UNIVERSIDAD COMPLUTENSE DE MADRID

FACULTAD DE INFORMÁTICA

DEPARTAMENTO DE ARQUITECTURA DE COMPUTADORES Y AUTOMÁTICA

TÉCNICAS DE DISEÑO PARA EL CONTROL DE LA TEMPERATURA EN ARQUITECTURAS MULTIPROCESADOR EN TRES DIMENSIONES

THERMAL AWARE DESIGN TECHNIQUES FOR MULTIPROCESSOR ARCHITECTURES IN THREE DIMENSIONS

TESIS DOCTORAL DE:
DAVID CUESTA GÓMEZ

DIRIGIDA POR:
JOSÉ LUIS AYALA RODRIGO
JOSÉ LUIS RISCO MARTÍN
JOSÉ IGNACIO HIDALGO PÉREZ

Madrid, 2014

Técnicas de Diseño para el Control de la Temperatura en Arquitecturas Multiprocesador en Tres Dimensiones

Thermal Aware Design Techniques for Multiprocessor Architectures in Three Dimensions

Tesis Doctoral

David Cuesta Gómez

Departamento de Arquitectura de Computadores y Automática
Facultad de Informática
Universidad Complutense de Madrid

2013

Técnicas de Diseño para el Control de la Temperatura en Arquitecturas Multiprocesador en Tres Dimensiones

Thermal Aware Design Techniques for Multiprocessor Architectures in Three Dimensions

Tesis presentada por
David Cuesta Gómez

Tesis dirigida por los doctores:
Dr. José Luis Ayala Rodrigo
Dr. José Luis Risco Martín
Dr. José Ignacio Hidalgo Pérez

Departamento de Arquitectura de Computadores y Automática
Facultad de Informática
Universidad Complutense de Madrid

2013

Técnicas de Diseño para el Control de la Temperatura en Arquitecturas Multiprocesador en Tres Dimensiones

Memoria presentada por David Cuesta Gómez para optar al grado de Doctor por la Universidad Complutense de Madrid, realizada bajo la dirección de Dr. José Luis Ayala Rodrigo, Dr. José Luis Risco Martín y Dr. Jose Ignacio Hidalgo Pérez (Departamento de Arquitectura de Computadores y Automática, Universidad Complutense de Madrid).

Thermal Aware Design Techniques for Multiprocessor Architectures in Three Dimensions

Report presented by David Cuesta Gómez to the Complutense University of Madrid in order to apply for the Doctor's degree. This work has been supervised by Dr. José Luis Ayala Rodrigo, Dr. José Luis Risco Martín and Dr. Jose Ignacio Hidalgo Pérez (Computers Architecture and Automation Department, Complutense University of Madrid).

Madrid, 2013

Este trabajo ha sido posible gracias a la Comisión Interministerial de Ciencia y Tecnologia, por las ayudas recibidas a través de los proyectos CICYT TIN2005/5619 y CICYT TIN2008/00508

A los que me dieron la vida
... y a los que daría la vida por ellos

Agradecimientos

Es de bien nacido el ser agradecido
Anónimo

Agradecer es sentir gratitud, que según la Real Academia de la Lengua es un sentimiento que nos obliga a estimar el beneficio o favor que se nos ha hecho o ha querido hacer, y a corresponder a él de alguna manera. En estos momentos no me siento obligado, lo hago porque quiero.

En primer lugar he de agradecer de una forma muy especial a mis tres directores, mis tres Joses, que han hecho que este sueño sea verdaderamente posible. Desde sus diferentes prismas y experiencia me han sabido llevar y asesorar hasta este momento en el que me veo. Creo que nunca podré agradecer de forma suficiente el esfuerzo que mis directores han realizado.

D. Jose Ignacio *AKA* Iñaki, me inoculó el gusanillo de la arquitectura de computadores. Él puso la primera piedra del proyecto y me ofreció grandes posibilidades de trabajo desde la asignatura que me impartió durante mis estudios de ingeniería, hasta ahora. Iñaki ha supervisado mi trabajo mostrando siempre un verdadero interés por el mismo.

D. Jose Luis Risco *AKA* Josele, el experto en algorimos genéticos, ha sido mi instructor matemático, algortímico, implementador, desarrollador de ideas. Sin ninguna duda sin su apoyo y esfuerzo esta tésis nunca podría haberse hecho.

Y por último pero no menos importante D. Jose Luis Ayala o simplemente Jose. Jose ha sido el director de la orquesta, el experto en arquitectura y tecnología. Su espíritu innovador y su gran conocimiento del mundo de la investigación ha hecho posible que este trabajo estuviera relacionado con otros de otras instituciones europeas.

Gracias a mis tres magníficos directores el conjunto ha funcionado como un engranaje de precisión, a pesar de que en el periodo de realización de este trabajo he tenido demasiadas "*distracciones*"y mi dedicación al mismo no ha podido ser tan intensa como hubiera deseado. Por ello también he de dejar escapar un enorme GRACIAS por entenderlo.

Sin embargo esta primera sección no debería escribirse íntegramente en español, debería aparecer al menos un GRAZIE MILLE. Gracias a mis com-

pañeros y amigos que conocí durante mi estancia de tan sólo seis meses en el Politecnico di Torino. En ese periodo inicié mi investigación en políticas de diseño térmico, el germen de lo que hoy constituye este trabajo. Me gustaría reconocer la hospitalidad y apoyo de todos mis compañeros, en especial el acogimiento de Alberto Bonanno, Karthik y Santa. Con ellos además de una gran experiencia profesional he compartido una gran experiencia humana.

Aunque mi periodo de becario FPU en la Universidad ha sido más breve de lo que cabría imaginarse he hecho muchos amigos y compañeros. A pesar de que nunca haya compartido realmente despacho ni laboratorio con ellos hemos coincido y preparado otros buenos momentos universiatarios, comidas, salidas nocturnas, celebraciones varias y un largo etcétera de actividades lúdico-recreativas. Ellos, han ido haciendo que la llama de la vida universitaria y la lucha por la investigación aún no se haya apagado en mi. Gracias a Alberto, Pablo, Fran, Carlos, Guillermo, Juanan, Antonio, Lucas e Íñigo. Irremediablemente nuestros caminos se han ido apartando pero espero que nunca se pierda el contacto de amistad y profesional.

Sería un error el no agradecer a todos los que hicieron posible mi primera etapa investigadora en la sección de automática. En este periodo aprendí a *cacharrear* y di mis primeros pasos en el mundo de la investigación universitaria. Gracias a Carlos León por su paciencia y codirección en mi proyecto fin de carrera de electrónica.

Tampoco puedo olvidarme de todas las personas que han contribuido a mi formación profesional, todos han aportado mucho, desde mi más odiada profesora de literatura del colegio, a mis directores de tesis. De un modo u otro todos han aportado su arena, ladrillos y demás material para construir mi proyecto de perfil profesional.

Y dejando de lado mi vida profesional, me centro en lo que es más importante para mi, mi vida social. Gracias a mis padres, hermano, primos, abuelos, a Belén y a la última incorporación al conjunto familiar, Raquel. Ellos son mi familia, mi alegría, mi apoyo y mi sustento. Muchas gracias a todos ellos con los que he vivido innumerables experiencias, me han criado y entendido en todo momento y por ello no puedo por más que dedicarles estas líneas.

Del mismo modo, a todos mis otros amigos no representados con anterioridad, con los que siempre he vivido el día a día, y con los que he llegado a ser el *friki* que compartía con ellos mis incertidumbres sobre el diseño de *chips 3D* y otras muchas ideas que me rondaban la cabeza.

En definitiva, muchas gracias a todos aquellos que estén leyendo estas líneas, porque si así lo han hecho es porque para ellos es importante y para mi en consecuencia, agradecer su interés. Seguramente esta sección, sea la más leída del libro que sigue, pero animo por lo menos a dejar pasar las páginas y mirar los dibujos como con los libros y cuentos de niños, al menos algo quedará.

Index

List of Figures

List of Tables

List of Algorithms

List of Acronyms

MPSoC Multi Processor System on Chip

3D 3 Dimensions

2D 2 Dimensions

GPU Graphics Processing Unit

TTSV Thermal Through Silicon Via

TSV Through Silicon Via

EPFL École Polytechnique Fédérale de Lausanne

ASIC Application Specific Integrated Circuit

VLSI Very Large Scale Integration

CAD Computer-Aided Design

IC Integrated Circuit

EDM Electro Static Discharge Machining

MEM Micro Electro Mechanic

LP Linear Programming

EA Evolutionary Algorithm

EDA Electronic Device Automation

SA Simulated Annealing

ILP Integer Linear Problem

MILP Mixed Integer Linear Problem

SA Simulated Annealing

GA Genetic Algorithms

PCB Printed Board Circuit

RC Resistance-Capacitance

DVFS Dynamic Voltage and Frequency Scaling

MOEA Multi-Objective Evolutionary Algorithm

NSGA Non Dominated Sorting Genetic Algorithm

VEGA Vector-Evaluated Genetic Algorithm

SPEA Strength Pareto Evolutionary Algorithm

PAES Pareto Archived Evolution Strategy

XML Extensible Markup Language

JECO Java Evolutionary Computation

CVD Chemical Vapor Deposition

PVD Physical Vapor Deposition

NOC Network on Chip

MFA Multiobjective Floorplanning Algorithm

TCG Transitive Closure Graph

SP Sequence Pair

GPE Generalized Polish Expression

Resumen

No hay que empezar siempre por la noción primera de las cosas que se estudian, sino por aquello que puede facilitar el aprendizaje.

Aristóteles

En cumplimiento del Artículo 4 de la normativa de la Universidad Complutense de Madrid que regula los estudios universitarios oficiales de postgrado, se presenta a continuación un resumen en español de la presente tesis, que incluye la introducción, objetivos, principales aportaciones y conclusiones del trabajo realizado.

0.1. Introducción

El proceso de escalado contínuo en las últimas decadas ha llevado consigo importantes mejoras en cuanto al tamaño y el rendimiento de los productos electrónicos, sin embargo ha abierto la puerta a numerosos desafíos de diseño como los problemas de comunicación y el control de la temperatura.

La tendencia de diseño hacia sistemas multi procesador ha sido posible gracias a los avances en las tecnologías de fabricación de semiconductores. Por otro lado, el incremento de la potencia disipada ha sido administrada mediante técnicas dinámicas como el *Dynamic Voltage and Frequency Scaling (DVFS)* [24], dominios de reloj [117] o políticas de migración de tareas [34]. Sin embargo todas estas técnicas en mayor o menor grado, tienen un impacto negativo en el rendimiento del sistema. Los enfoques en tiempo de diseño, como el que se propone en este trabajo, son capaces de mitigar el efecto de las altas temperaturas y además son compatibles con las políticas dinámicas existentes.

Un mecanismo importante a la hora de superar los límites físicos en el diseño de Circuitos Integrados (IC) radica en el diseño de circuitos multi nivel mediante el empleo de sofisticados y avanzados procesos tecnológicos. Estas técnicas permiten la creación de circuitos integrados en tres dimensiones (3D). Los diseños 3D mejoran el rendimiento del sistema mediante

la reducción de los retardos debidos a las longitudes de interconexión. La inclusión de *Through Silicon Vias (TSVs)* que conectan varias capas del sistema tridimensional permite la reducción de las distancias entre las unidades funcionales, lo que se traduce en un decremento del retraso debido a las comunicaciones. Esta tecnología de fabricación también permite la integración de diferentes tecnologías tales como radio frecuencia, dispositivos de entrada salida analógica, memorias, procesadores...

En contraposición, la integración 3D incrementa el problema de la temperatura en el chip, especialmente la temperatura de las capas internas. Estos problemas térmicos afectan cada vez más el rendimiento y la fiabilidad de los dispositivos electrónicos. Shang en su publicación [134], ya expuso que más del 50 % de los fallos de productos electrónicos son causados por problemas térmicos y la aparición de puntos calientes (hot spots). El incremento de la temperatura hace que exponencialmente los dispositivos funcionen más lentamente, puede incrementar las corrientes de fuga y puede reducir el rendimiento del dispositivo debido a la variación de la resistividad del metal.

Teniendo en cuenta lo expuesto anteriormente, se desea que los componentes del la estructura del chip permanezcan tan fríos como sea posible para maximizar la fiabilidad de los mismos. Sin embargo, la temperatura máxima del chip no es el único factor que afecta al rendimiento, también la aparición de gradientes térmicos en la superficie degrada la fiabilidad del sistema y puede producir electromigraciones.

Los diseñadores de hardware han intentado hacer frente al problema térmico mediante el uso de técnicas de diseño orientadas térmicamente como [29] que propuso un algoritmo de emplazamiento desde del punto de vista térmico para sistemas en 3D o [67], quien implementó un algoritmo de emplazamiento multi objetivo para 2D y 3D, combinando programación lineal y *simulated annealing*. Otros autores también han considerado la colocación de vías térmicas para optimizar el perfil térmico de los circuitos integrados [155]. La colocación cautelosa de los módulos activos en el volumen de diseño 3D puede optimizar de forma significativa la longitud de cableado que conecta las unidades funcionales y reducir de este modo los retardos debidos a la comunicación, al mismo tiempo que podría producir una distribución homogénea de la temperatura a los largo del chip. Además de las aproximaciones de diseño estáticas, también se pueden emplear técnicas dinámicas para adminisitrar las altas densidades de potencia. Se ha probado que el sistema de soplado de aire tradicional no es suficiente para aliviar el extrés térmico de los diseños 3D, sin embargo el empleo de sistemas alternativos como los micro canales líquidos que pueden mapearse entre capas pueden ofrecer mejores resultados. Trabajos como [149] y [32] se han centrado en el modelado de sistemas de enfriamiento activo. Estos trabajos han estudiado el efecto de la colocación de canales líquidos entre capas activas.

Algunos de los objetivos del diseño de chips 3D son los de minimizar el

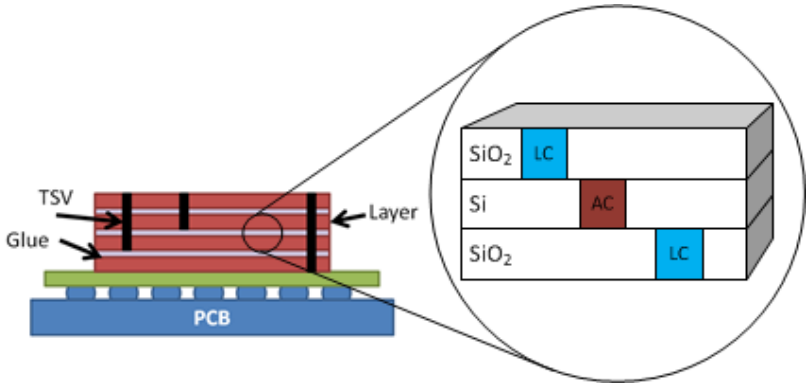


Figura 1: Concepto arquitectónico 3D

Functional Units	Air Channels	TSVs	Liquid Channels
Input: • Processors, memories, crossbars, set of connections between components, max. chip dimensions, etc. Algorithms: • MFA_{FU} or $MFA_{FU'}$	Possible inputs: • Processors, memories, crossbars, set of connections between components, max. chip dimensions, etc. • MFA_{FU} output • $MFA_{FU'}$ output Algorithm: • MFA_{AC}	Possible inputs: • MFA_{FU} output • $MFA_{FU'}$ output • MFA_{AC} output • Any feasible design Algorithm: • MFA_{TSV}	Possible inputs: • Any feasible design • MFA_{FU} output • $MFA_{FU'}$ output • MFA_{AC} output • MFA_{TSV} output • Any feasible design Algorithm: • MFA_{LC}

Figura 2: Flujo de diseño

área y por lo tanto disminuir la longitud de las intercoxiones, que podrían ser trasladadas en un decremento del tiempo de transferencia de datos y la potencia asociada a este hecho. La Figura 1 resume el concepto de propuesta de la arquitectura 3D. Se basa en el modelo propuesto por [129]. El chip se construye sobre un circuito impreso que se considera adiabático. Posteriormente diferentes capas se sitúan unas encima de otras, como puede verse en la Figura 1. Cada una de las capas se compone de silicio y de dióxido de silicio. Los canales líquidos (LC) se usan como un sistema de enfriamiento activo. Éstos, se colocan en la capa de dióxido de silicio inmediatamente superior a las capas activas, y absorben el calor producido por las unidades funcionales con gran disipación de potencia, reduciendo de este modo la temperatura de las capas internas. Los canales contienen un líquido, generalmente agua, que es inyectado al interior del chip.

En este trabajo además de las técnicas descritas con anterioridad se propone un novedoso diseño de arquitectura basado en la inclusión de canales de aislamiento de aire (AC) Los canales de aire se mapean en la capa activa y son rellenados de aire a baja presión. Dado que el mecanismo de difusión del calor es mayormente difusivo, los canales de aire impiden que las áreas frías se vean afectadas térmicamente por aquellas con mayor temperatura, creando de esta forma dominios térmicos.

Todos estos puntos hacen necesario que se abra una línea de investigación en el diseño e implementación de algoritmos de optimización para colocar las unidades funcionales, las TSVs, los canales líquidos y loa canales de aislamiento de aire, minimizando la temperatura y la longitud total de cableado. La administración eficiente de todos estas variables de entrada, así como los múltiples criterios de optimización y los diferentes objetivos, hacen que sea necesario el desarrollo de algoritmos innovadores capaces de evaluar todos los parámetros y devolver el mejor conjunto de soluciones posible. Para ello en este trabajo se construye un algoritmo llamado *Multi-objective Floorplanning Algorithm (MFA)* [37]. El MFA permite una colocación incremental de las unidades funcionales. Como se muestra en la Figura 2, en el flujo de optimización se integra la colocación óptima de TSVs, canales líquidos y canales de aire.

0.2. Modelo Térmico

La ecuación que gobierna la difusión del calor por coducción es bien conocida y se puede describir como:

$$\rho c \frac{\partial T(\vec{r},t)}{\partial t} = \nabla \left(k(\vec{r}) \nabla T(\vec{r},t) \right) + p(\vec{r},t) \tag{1}$$

Sujeta a las condiciones de contorno del chip quedaría:

$$k(\vec{r},t) \frac{\partial T(\vec{r},t)}{\partial n_i} + h_i T(\vec{r},t) = f_i(\vec{r},t) \tag{2}$$

En estas ecuaciones ρ es la densidad del material, c es la capacidad térmica en masa, $T(\vec{r},t)$ y $k(\vec{r})$ son la temperatura y la conductividad térmica del material en la posición $\vec{r}$ y en el instante t, y $p(\vec{r},t)$ es la densidad de potencia de la fuente de calor. n_i define la dirección normal a la superficie i, h_i es el coeficiente de transmisión del calor y f_i es un valor de la función en la superficie i.

Para llevar a cabo el análisis numérico y por ende, térmico, se puede aplicar un método discretización de diferencias finitas en siete puntos 1, de modo que la Ecuación 1, se pueda analizar mediante la descomposición del chip 3D en numerosos paralelepípedos rectangulares. De este modo, cada elemento en el que se descompone la estructura, tendrá una disipación de potencia, temperatura, capacidades y resistividades térmicas características. La ecuación discretizada en un punto interior de la malla seguiría la relación:

$$\begin{aligned}\rho cV\frac{T^{q+1}_{i,j,l}-T^{q}_{i,j,l}}{\Delta t} &= -2(R_x+R_y+R_z)T^{q}_{i,j,l}+R_xT^{q}_{i-1,j,l}\\ &\quad +R_xT^{q}_{i+1,j,l}+R_yT^{q}_{i,j-1,l}\\ &\quad +R_yT^{q}_{i,j+1,l}+R_zT^{q}_{i,j,l-1}\\ &\quad +R_zT^{q}_{i,j,l+1}+Vp_{i,j,l} \qquad (3)\end{aligned}$$

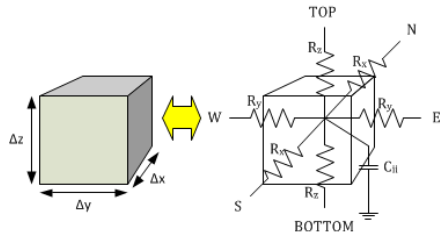
donde i, j y l son las posiciones en los ejes x, y y z, Δt es el valor de discretización en el tiempo t, Δx,Δy y Δz las discretizaciones en los ejes. Finalmente R_x, R_y y R_z son las conductividades térmicas entre bloques vecinos que se define como

$R_x = k\frac{\Delta_y\Delta_z}{\Delta_x}$, $R_y = k\frac{\Delta_x\Delta_z}{\Delta_y}$, y $R_z = k\frac{\Delta_x\Delta_y}{\Delta_z}$.

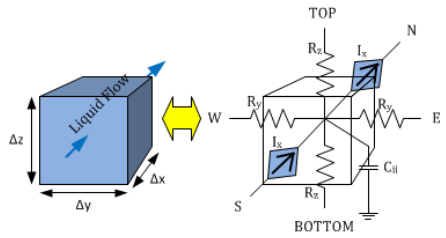
En un chip 3D con una discretización de N elementos, la ecuación 3 se puede resumir en:

$$\mathbf{C}\frac{dT(t)}{dt}+\mathbf{R}T(t)=Pu(t) \qquad (4)$$

donde la matriz de capacidades térmicas $\mathbf{C}$ es una matriz diagonal $N\times N$, la matriz de conductividad $\mathbf{R}$ es una matriz de $N\times N$, $T(t)$ y P son vectores de $N\times 1$ que definen la temperatura y la potencia respectivamente y $u(t)$ es la unidad de tiempo.



(a)



(b)

Figura 3: Celdas de descomposición: (a) Celda difusiva (silicio, dióxido de silicio, aire) and (b) Celda de líquido.

La ecuación (4) puede ser modelada mediante un sistema de resistencias y capacidades (RC) como el presentado en [9]. Las diferencias de voltaje son

análogas a las diferencias de temperatura, y la resistencia eléctrica es análoga a la resistencia térmica. El modelado térmico del chip tridimensional que se muestra en la Figura 1, se realiza después de haber dividido la estructura entera en pequeñas celdas unitarias.

Las celdas de silicio y de dióxido de silicio se modelan con seis resistencias eléctricas y un condensador como se puede apreciar en la Figura 3a. Cuatro de las resistencias conectan a la celda con sus vecinas laterales que se encuentran en la misma capa, mientras que las dos restantes conectan la celda con sus vecinas superior e inferior. El condensador representa el almacenamiento de calor de la propia celda.

Los canales de aire, tal y como ocurre con las celdas sólidas tienen un comportamiento difusivo. A pesar del hecho de que el aire es un fluido, las dimensiones de la cavidad y la ausencia de métodos de convección forzados, hacen que el fluido se comporte como un material difusivo. En este trabajo se han considerado canales rellenos con aire a baja presión, lo que decrece la conductividad térmica del material haciendo que el aislamiento que éste proporciona sea mucho más eficiente.

Los canales líquidos están modelados eléctricamente como se muestra en la Figura 3b. La diferencia con las celdas difusivas radica en el hecho de que por estas celdas circula un refrigerante, de este modo, el mecanismo de conducción térmica que domina es la convección forzada. Este mecanismo se traduce en nuestro modelo térmico mediante el uso de dos fuentes de tensión controladas por corriente, tal y como se describe detalladamente en [140].

La difusión del calor hacia el ambiente también está considerada en nuestro modelo RC. Esta difusión de calor se da siempre en los extremos del chip y en la capa superior. Mediante el ajuste de los valores de capacidad y de resistencias en las celdas limítrofes, se pueden modelar diferentes empaquetados con características térmicas diferentes. Como el circuito impreso que sirve como base se considera adiabático no existe transferencia de calor en esta dirección desde la primera capa.

El material adherente que existe entre las dos capas se modela como un material que es puramente resistivo. El modelo también incluye la existencia de TSVs que también son consideradas como un material puramente resistivo.

Todos estos tipos de celdas se integran en la Ecuación (4), que se resuelve mediante el uso de un método iterativo (Forward Euler) para validar los resultados devueltos por nuestro optimizador. Las principales propiedades térmicas de los materiales usados así como sus dimensiones geométricas se listan en las Tablas 1 y 2.

Término lineal de la conductividad del Si	295 W/(mK)
Término cuadrático de la conductividad del Si	-0.491 W/(mK^2)
Conductividad térmica del SiO_2	1.38 W/(mK)
Calor específico del Si	1.628 x 10^6 J/m^3K
Calor específico del SiO_2	4.180 x 10^6 J/m^3K
Conductividad térmica del aire aislante	2.4 x 10^-3 W/(mK)
Calor específico del aire aislante	1 x 10^4 J/m^3K
Calor específico del agua	4.184 x 10^6 J/m^3K
Conductividad térmica del agua	0.58 W/(mK)

Tabla 1: Propiedades térmicas de los materiales.

Anchura del chip	12000 μm
Profundidad del chip	10500 μm
Tamaño de la celda(lxw)	300x300 μm
Tamaño de la celda de una TSV (lxw)	300x300 μm
Anchura del canal líquido	300 μm
Altura de la capa de SiO_2	50 μm
Altura de la capa de Si	150 μm
Altura de la capa de Epoxy	25 μm

Tabla 2: Propiedades geométricas del chip..

0.3. Flujo de diseño: Algoritmo de emplazamiento multiobjetivo (MFA)

El MFA es un algotimo de emplazamiento que se desarrolla en dos etapas. En la primera fase el algoritmo se ejecuta para colocar las unidades funcionales. A esta fase del algoritmo la denotaremos como MFA_{FU}. Seguidamente la segunda fase MFA_{TSV} se ejecuta para colocar las TSVs. Estos dos procesos son independientes, aunque como veremos seguidamente no lo son totalmente.

0.3.1. MFA_{FU}

MFA_{FU} es un *Multi-Objective Evolutionary Algorithm (MOEA)* basado en NSGA-II [44]. El optimizador maneja soluciones codificadas que son continuamente mejoradas por el proceso evolutivo para finalmente proveer soluciones optimizadas en rendimiento y temperatura.

El MFA_{FU}, tiene tres objetivos que tienen que ser minimizados simultáneamente:

- F_1: El número de restricciones topológicas. No puede existir solapamiento en la colocación de las unidades funcionales, y éstas tampoco

pueden sobrepasar los límites del chip.

- F_2: Longitud de cableado, aproximado mediante la distancia Manhattan entre bloques conectados $\mathbf{C}$ de la siguiente forma:

$$F_2 = \sum_{i,j \in \mathbf{C}: i<j} |x_i - x_j| + |y_i - y_j| + |z_i - z_j| \tag{5}$$

  , donde (x_i, y_i, z_j) son las coordenadas de la unidad funcional i.

- F_3: Temperatura máxima del chip. El cálculo de esta métrica depende del modelo térmico. En el caso del MFA_{FU}, se aproxima este valor teniendo en cuenta que la contribución a la temperatura máxima de dos unidades funcional i, j que se simplifica como el producto cruzado de sus densidades de potencia p_i, p_j dividida por su correspondiente distancia euclídea. Por tanto, teniendo n unidades funcionales, el objetivo F_3 se define como:

$$F_3 = \sum_{i<j \in 1...n} \frac{p_i \cdot p_j}{\sqrt{(x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2}} \tag{6}$$

  Al minimizar el objetivo F_3, el algoritmo intenta colocar las unidades funcionales más calientes tan lejos como sea posible las unas de las otras. Este proceso reduce la temperatura mínima como se demuestra en [37].

El MFA_{FU} puede clasificarse como un optimizador híbrido ya que el decodificado heurístico es implementado como un optimizador incremental inspirado en técnicas de construcción, mientras que el MOEA que se coloca por encima de la heurística es esencialmente iterativo.

Para codificar las soluciones se emplea una codificación de permutación [138], en el que todos los cromosomas son una cadena de registros que representan las diferentes unidades funcionales caracterizadas en la arquitectura. Estos registros reúnen la información relativa a las unidades funcionales, la etiqueta de nombre, anchura y altura. Mediante la gestión de la anchura y altura se pueden efectuar rotaciones, añadiendo grados de libertad al proceso de optimización. Las características adicionales de las unidades funcionales como las densidades de potencia, conexiones, etc., también tienen que ser manejadas por el algoritmo. Sin embargo, no es necesario que esta información se codifique en los cromosomas ya que es común a los individuos. La Figura 4a muestra la representación de un cromosoma usado por el MFA_{FU}. El ejemplo muestra una solución candidata (individuo) de un simple sistema completo compuesto por 8 unidades funcionales: 4 procesadores $C_i (i = 1, 2, 3, 4)$, y 4 memorias $M_i (i = 1, 2, 3, 4)$. El orden en el que aparecen las unidades funcionales en el individuo C1,C2,M1,M2,C3,M3,M4,C4;

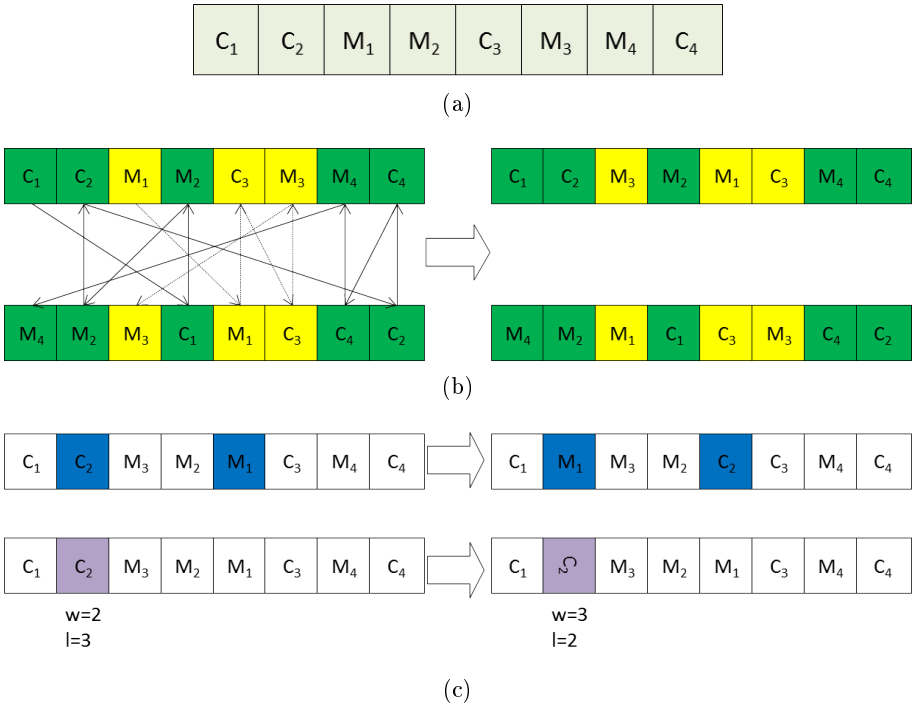


Figura 4: Cromosoma del MFA_{FU} (a), operador de cruce (b) y mutación (c).

determina la secuencia de emplazamiento. Por lo tanto, la unidad C1 será la primera en ser colocada, seguida del procesador C2 y así sucesivamente.

Es lógico pensar que todos los cromosomas tienen que ser de tamaño n, donde n representa el número de unidades funcionales que compone el sistema completo. Por tanto la cardinalidad del espacio de soluciones considerado es de $n!$. Los operadores evolutivos diseñados para esta representación se muestran en la Figura 4 y se describen brevemente a continuación:

- *Selección:* El operador de selección se implementa mediante una estrategia de torneo binario. Para ello, se escogen de forma aleatoria pares de individuos y la mejor solución de cada par es seleccionada.

- *Cruce:* El ciclo de cruce se usa para producir los hijos. Este operador tiene que tener en cuenta que todos los componentes deben de aparecer una sóla vez en el cromosoma, para ello se implementa el proceso que se muestra en la Figura 4b.

- *Mutación:* La mutación de las soluciones se puede dar de dos formas diferentes. La primera consistiría en cambiar la posición de dos unidades funcionales en el cromosoma, resultando en un cambio de localización de la secuencia de emplazamiento de los individuos mutados. La segunda forma de mutación consiste en la rotación de los elementos (ver

Figura 4c).

El algoritmo 0.1 muestra la implementación del MFA_{FU}. Una heurística sofisticada se encarga de la colocación de los diferentes elementos arquitectónicos (decodificación de las soluciones). Esta heurística desarrolla un emplazamiento incremental en la que los componentes se colocan secuencialmente en el volumen de diseño, siguiendo el orden codificado en la solución. De hecho, la heurística por si sola, sería capaz de encontrar soluciones, el MOEA se ha diseñado por debajo de ésta, para que las soluciones sean óptimas. En la heurística cada bloque i se coloca teniendo en cuenta todas las restricciones topológicas, la longitud de cables y la temperatura máxima del chip respecto a los bloques previamente colocados $j : j < i$, nombrados como (F_1^i, F_2^i, F_3^i). La mejor posición de cada bloque se selecciona dependiendo de si el bloque es considerado como una fuente de calor o como un sumidero. Si es una fuente de calor, como por ejemplo un procesador, el mejor punto para colocarlo sería uno con el menor F_3^i, ya que así se aseguraría una buena distribución térmica. Si el bloque es considerado como un sumidero térmico, como por ejemplo una memoria, el mejor punto es aquel que minimice la longitud de cable F_2^i. Con este procedimiento, se trata de obtener una optimización del comportamiento térmico del diseño. El enfoque es perfectamente razonable en términos de perfiles térmicos, dado que grandes diseños con más de 48 procesadores podrían alcanzar temperaturas descabelladas por encima de los 400 K como se muestra en [37]). Por lo tanto, cada bloque se va fijando en una posición fija hasta que todas las unidades funcionales hayan sido ubicadas en el volumen de diseño. Una vez se ha realizado este proceso, la configuración final es valorada de acuerdo con los tres objetivos (F_1, F_2, F_3).

0.3.2. Optimización de TSVs: MFA_{TSV}

Como se ha dicho anteriormente, este algoritmo es el responsable de la colocación de las TSVs que permiten la comunicación vertical entre diferentes capas. Tal y como sucede en el MFA_{FU}, el MFA_{TSV} está basado en un NSGA-II. En la versión original del MFA, este algoritmo se ejecutaba inmediatamente después del anterior sin comprobar que la inserción de TSVs en la solución fuera posible. Sin embargo, como se explicará posteriormente, el MFA_{FU} ha sido mejorado para asegurar que se puedan introducir TSVs en el diseño y garantizar la comunicación vertical.

Tecnológicamente, y debido a las restricciones en el proceso de fabricación, las TSVs sólo pueden conectar dos capas diferentes.

Para codificar una solución, el MFA_{TSV} examina las celdas que no están ocupadas por ninguna unidad funcional colocada por el algoritmo anterior MFA_{FU}. Posteriormente, y con los datos recopilados de la comprobación, el MFA_{TSV}, construye un vector de coordenadas x-y de posiciones susceptibles

Algorithm 0.1 MFA_{FU}

Require: G es el número de generaciones. N es el tamaño de la población.

function main()
P = initialize() {P es la primera población aleatoria}
evaluate(P) {P es evaluada}
for $g = 1$ **to** G **do**
  $\hat{P} = \emptyset$ {Nueva población vacía}
  for $n = 1$ **to** $N/2$ **do**
    $\hat{P_s}$ = select(P) {Selección de dos individuos,}
    $\hat{P_c}$ = crossover($\hat{P_s}$) {ejecución de cruce ...}
    $\hat{P_m}$ = mutation($\hat{P_c}$) {y mutación}
    $\hat{P} = \hat{P} \cup \hat{P_m}$
  end for
  evaluate($\hat{P}$)
  $P = P \cup \hat{P}$
  reduce(P) {Mecanismo standard de reducción para NSGA-II}
end for

function evaluate(P)
for all $I \in P$ **do**
  for $i = 1$ **to** n **do**
    $B_i \leftarrow I_i$ {El gen i-ésimo del individuo I (I_i) representa el bloque i-ésimo/unidad funcional (B_i) que debe ser colocada en el diseño 3D candidato}
    $f_i^* \leftarrow \infty$, $x_i^* \leftarrow 0, y_i^* \leftarrow 0, z_i^* \leftarrow 0$
    for all $(x_i \in [0..L - l_i], y_i \in [0..W - w_i], z_i \in [0..H - h_i])$ **do**
      $F_1^i \leftarrow$ checkTopologyConstraints(x_i, y_i, z_i, i) {Número de restricciones topológicas violadas con los anteriores $i-1$ bloques emplazados}
      if $F_1^i = 0$ **then**
        $F_2^i \leftarrow$ manhattan(x_i, y_i, z_i, i) {Cálculo de la longitud de cable de acuerdo a las distancias de Manhattan entre los bloques conectados en el rango $[1..i]$}
        $F_3^i \leftarrow$ computeTemp(x_i, y_i, z_i, i) {Cálculo de (6) con $i < j \in 1..i$}
        if B_i es un procesador **then**
          $f_i \leftarrow F_3^i$
        else
          $f_i \leftarrow F_2^i$
        end if
        if $f_i < f_i^*$ **then**
          $f_i^* \leftarrow f_i$
          $x_i^* \leftarrow x_i, y_i^* \leftarrow y_i, z_i^* \leftarrow z_i$
        end if
      end if
    end for
    $B_i \leftarrow (x_i^*, y_i^*, z_i^*)$ {Asignar las mejores coordenadas a cada bloque}
  end for
  $F_1 \leftarrow$ checkTopologyConstraints() {Número de restricciónes topológicas violadas en el actual diseño 3D}
  $F_2 \leftarrow$ manhattan() {Longitud total de cable}
  $F_3 \leftarrow$ computeTemp() {Cálculo de (6)}
  $I \leftarrow (F_1, F_2, F_3)$ {Asignación de los valores de los objetivos a cada individuo}
end for

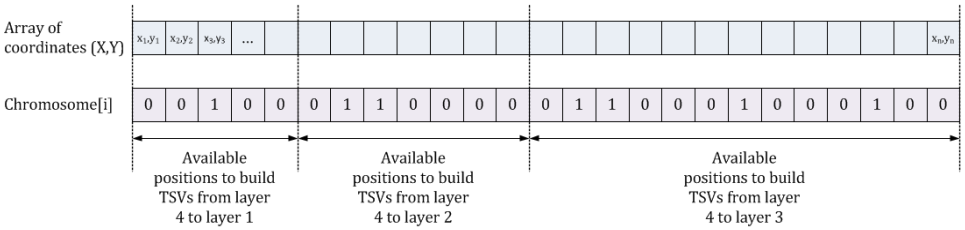


Figura 5: Descripción del cromosoma de colocación de TSVs.

de ser atravesadas por una TSV, tal y como representa la Figura 5. Dado un diseño 3D con N capas, el primer conjunto muestra las regiones donde se pueden insertar TSVs entre las capas Top y 1, un segundo conjunto donde se pueden insertar TSVs entre las capas Top y 2 y así sucesivamente. Posteriormente, se crea un cromosoma de 0 y 1 de longitud idéntica al vector anterior. Si la posición del cromosoma (gen) contiene un 1, indica que existe una TSV en dicha posición (x,y) entre las capas que se corresponden con el conjunto en cuestión. De este modo en la Figura 5 se codifican 7 TSVs en cuatro capas ($N = 4$): 1 TSV que se inserta entre las capas 4 y 1, 2 TSVs entre las capas 4 y 2 y 4 TSVs entre las capas 4 y 3. Las coordenadas correspondientes se almacenan en el vector de coordenadas.

El Algoritmo 0.2, muestra el pseudocódigo del MFA_{TSV}. Este algoritmo devuelve un conjunto de soluciones, que tiene en cuenta el número de TSVs F_4 y la longitud total de cableado F_5. Este conjunto constituye una aproximación de frente Paret, y será el diseñador el que decida la mejor solución en términos de costes económicos y reducción en la longitud de cable, teniendo en cuenta que se debe de insertar un mínimo de TSVs para cumplir con los requisitos de comunicación. El número mínimo de TSVs se calcula considerando la necesidad de ancho de banda entre los procesadores. Ésta se ha calculado haciendo un estudio de una aplicación de modulación/demodulación como la explicada en [34]. El número mínimo de TSVs está dado por los parámetro tecnológicos de las mismas y la cantidad de datos que se tienen que transferir [51].

0.3.3. $\text{MFA}_{\text{FU}*}$

En esta parte, como ya se introdujo con anterioridad, se modifica el algoritmo MFA_{FU} original para poder obtener soluciones donde, al menos se pueda alcanzar una configuración de TSVs viable. Para ello se propone el Algoritmo 0.3. En este algoritmo sólo se mostrará la función de evaluación ya que la función principal es exactamente igual al MFA_{FU}. Para ello se incluye en la descripción la matrix R. R contiene todas las celdas libres del volumen de diseño. Por lo tanto, dado un diseño, es sencillo comprobar si una TSV se puede incorporar para conectar dos unidades funcionales situadas en

Algorithm 0.2 MFA_{TSV}

Require: I es el individuo actual (ver Figura 4).C es el conjunto de conexiones del floorplan.

function evaluate(I)
$F_4 \leftarrow \sum_{i=1..N} I_i$ {Número de TSVs.}
$F_5 \leftarrow 0$
for all $(B_i, B_j) \in C$ **do**
  $dz \leftarrow |z_i - z_j|$
  if $dz = 0$ **then**
    $d \leftarrow$ manhattan(x_i, y_i, x_j, y_j) {Distancia de Manhattan entro los bloques i y j}
    $F_5 \leftarrow F_5 + d$
  else
    $d \leftarrow$ findBestTsv(I, i, j) {Esta función coge todos las TSVs candidatas en I y calcula la distancia Manhattan del conexionado: bloque $i \rightarrow$ TSV $\rightarrow$ bloque j. Al final, devuelve la mejor distancia posible.}
    $F_5 \leftarrow F_5 + d$
  end if
end for
$I \leftarrow (F_4, F_5)$ {Asigna los valores multi objetivos a este individuo}

diferentes capas. Si no es posible, el valor del objetivo F_2^i adquiere el valor infinito, descartando la opción de diseño por ser éste inviable.

A continuación se presentarán dos nuevos subalgoritmos del MFA llamados MFA_{LC} y MFA_{AC}. El primero de ellos se ha desarrollado para colocar los canales líquidos en el diseño del chip 3D. El segundo, ha sido diseñado para dividir el volumen de diseño en regiones aisladas por canales de aire. Estos dos algoritmos reciben como entrada el circuito 3D devuelto por MFA_{FU} o por el el $\text{MFA}_{\text{FU}*}$+MFA_{TSV}, tal y como refleja la Figura 2.

0.3.4. Optimización de los canales líquidos: MFA_{LC}

Una vez que el diseño 3D ha sido optimizado desde el punto de vista térmico al colocar las unidades funcionales o las TSVs, se ejecuta el modelo térmico descrito en la ecuación (4), de modo que se obtengan todas las temperaturas de las celdas en las que el volumen de diseño había sido descompuesto. Estos valores, junto con la información del diseño sirven como entrada al optimizador de canales de aire como se muestra en el Algoritmo 0.4.

Dado que los canales líquidos se colocan justo encima de las unidades funcionales, la única restricción topológica es la posible colisión entre un canal líquido y una TSV. Tal y como se ha hecho previamente en la colocación de las TSVs, se construye un conjunto de coordenadas disponibles (x, z). La función de evaluación del MFA_{LC}, toma en cuenta el efecto de enfriado que tiene un canal líquido. Para evaluar este efecto, se realizaron varias simulaciones. De ellas se desprende que los canales líquidos no sólo son capaces de enfriar las celdas que se sitúan justo debajo de ellos, sino que también son

Algorithm 0.3 $\text{MFA}_{\text{FU}*}$

Require: G es el número de generaciones. N es el tamaño de la población.

function evaluate(P)
for all $I \in P$ **do**
$R \leftarrow 1$ {Matriz $L \times W \times H$ con las celdas libres en el 3D-IC, donde se pueden colocar las unidades funcionales.}
for $i = 1$ **to** n **do**
$B_i \leftarrow I_i$ {El i-ésimo gen del individuo I (I_i) que representa al bloque i-ésimo (B_i) para ser colocada en el actual diseño candidato}
$f_i^* \leftarrow \infty$, $x_i^* \leftarrow 0, y_i^* \leftarrow 0, z_i^* \leftarrow 0$
for all $(x_i \in [0..L - l_i], y_i \in [0..W - w_i], z_i \in [0..H - h_i])$ **do**
$F_1^i \leftarrow$ checkTopologyConstraints(x_i, y_i, z_i, i) {Número de restricciones topológicas violadas por los $i - 1$ bloques colocados previamente}
if $F_1^i = 0$ **then**
$F_2^i \leftarrow$ manhattan(x_i, y_i, z_i, i, R) {Esta función calcula la longitud de cable de acuerdo a la distancia Manhattan entre las unidades conectadas en el rango $[1..i]$. Igualmente informa de que una TSV se tiene que crear si dos de estos bloques están en diferentes capas.}
$F_3^i \leftarrow$ computeTemp(x_i, y_i, z_i, i) {Calcula (6) con $i < j \in 1..i$}
if B_i es un procesador **then**
$f_i \leftarrow F_3^i$
else
$f_i \leftarrow F_2^i$
end if
if $f_i < f_i^*$ **then**
$f_i^* \leftarrow f_i$
$x_i^* \leftarrow x_i, y_i^* \leftarrow y_i, z_i^* \leftarrow z_i$
end if
end if
end for
$B_i \leftarrow (x_i^*, y_i^*, z_i^*)$ {Asigna las mejores coordenadas para cada bloque}
update(R, B_i)
end for
$F_1 \leftarrow$ checkTopologyConstraints() {Número de restricciones topológicas violadas en el diseño 3D}
$F_2 \leftarrow$ manhattan() {Longitud total de cable}
$F_3 \leftarrow$ computeTemp() {Cálculo (6)}
$I \leftarrow (F_1, F_2, F_3)$ {Asignación de valores multi objetivo para cada individuo}
end for

capaces de enfriar las vecinas. Esta reducción de la temperatura de la capa activa sigue una tendencia logarítmica como puede verse en la Figura 6. Al introducir estos valores en la función de evaluación del algoritmo podemos encontrar las posiciones óptimas de los canales líquidos.

El optimizar el número de micro canales en el diseño es extremadamente importante desde el punto de vista de los costes tecnológicos y de operación. El añadir más micro canales implica no sólo costes en el proceso de fabricación del chip, sino que también hay que considerar el consumo de potencia del sistema de bombeo, como se muestra en [93].

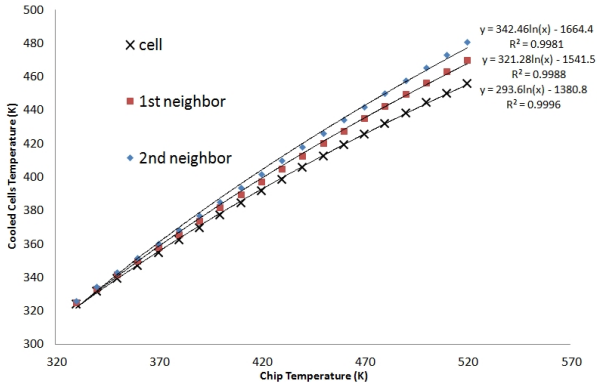


Figura 6: Regresión para la evaluación del efecto de canales líquidos en el proceso de optimización

Algorithm 0.4 MFA_{LC}

Require: I es el individuo actual para ser evaluado. T es el conjunto de temperaturas devuelto por el modelo térmico.

function evaluate(I)
$F_6 \leftarrow \sum_{i=1..N} I_i$ {Número de canales líquidos.}
$F_7 \leftarrow 0$
$\hat{T} \leftarrow T$
for all $I_i \in I$ **do**
  if $I_i = 1$ **then**
    $(x_i, z_i) \leftarrow I_i$ {Cada gen se refiere a una posición concreta (x_i, z_i), donde z_i es la capa del canal i-ésimo y x_i es su x coordenada}
    for $y_i = 0$ **to** $W - 1$ **do**
      $\hat{T}(x_i, y_i, z_i) = 342.46ln(\hat{T}(x_i, y_i, z_i)) - 1664.4$
      $\hat{T}(x_i - 1, y_i, z_i) = 321.28ln(\hat{T}(x_i - 1, y_i, z_i)) - 1541.5$
      $\hat{T}(x_i - 2, y_i, z_i) = 293.60ln(\hat{T}(x_i - 2, y_i, z_i)) - 1380.8$
      $\hat{T}(x_i + 1, y_i, z_i) = 321.28ln(\hat{T}(x_i + 1, y_i, z_i)) - 1541.5$
      $\hat{T}(x_i + 2, y_i, z_i) = 293.60ln(\hat{T}(x_i + 2, y_i, z_i)) - 1380.8$
    end for
  end if
end for
$F_7 = \sum_{x_i, y_i, z_i} \hat{T}((x_i, y_i, z_i))$
$I \leftarrow (F_6, F_7)$ {Se asignan los valores de los objetivos al individuo}

0.3.5. Colocación con canales de aire: MFA_{AC}

La propuesta de crear dominios térmicos en el chip es un método revolucionario para mantener el calor concentrado en ciertas zonas, impidiendo, mediante canales de aislamiento de aire, la difusión del calor desde zonas calientes a frias. Podría parecer extraño, pero si esta técnica se combina con los canales líquidos, se consiguen mejores resultados en términos de ahorro de energía y de costes de fabricación.

Dado que la conductividad térmica del aire es menor que la del silicio,

los canales pueden crear dominios térmicos aislando las regiones con altas temperaturas de otras zonas del chip. Con este escenario se puede obtener una distribución térmica más homogénea con una inversión en sistemas activos de enfriado. La creación de dominios térmicos, implica una reducción en el número de canales líquidos y en consecuencia una reducción en los costes de fabricación y operación.

Cuando el chip se encuentra aislado por canales de aire, el proceso de optimización para la colocación de las unidades funcionales está guiado. El proceso funciona como el descrito anteriormente, pero restringiendo ciertas áreas del chip a ciertas unidades funcionales dependiendo de su consumo de potencia. Las áreas calientes se compondrían de unidades funcionales con una alta densidad de potencia. Por otro lado, los elementos con un consumo menor, serían colocados, todos juntos, en regiones frías.

El algoritmo considerado es exactamente igual que el Algoritmo 0.1 o que 0.3, pero se añaden restricciones topológicas adicionales en la función `checkTopologyConstraints`. Obviamente, ninguna unidad funcional ni TSV puede ocupar los espacios destinados a los canales de aire.

La definición de la topología la da el diseñador y en consecuencia, es una entrada al sistema de optimización tal y como se ve en la Figura 2.

0.4. Escenario Experimental

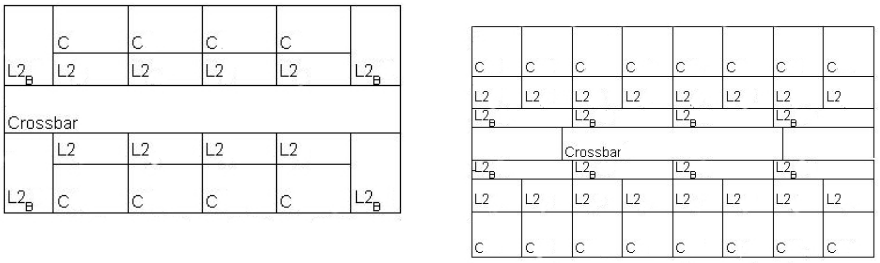
Las arquitecturas que se han tomado como base para la confección de nuestro escenario de pruebas son las del Niagara2 y el Niagara3. Estas distribuciones han sido modificadas para incluir 48 procesadores SPARC fabricados con tecnología de 90 nm.

Los 48 procesadores se han distribuido en 4 capas, compuestas de 8 procesadores originales del Niagara2 en las capas uno y dos, y 16 procesadores del Niagara3 en las capas tres y cuatro. Este escenario será el que sea empleado para todas las optimizaciones y simulaciones.

Se ha seleccionado el peor caso en cuestión de consumo de potencia. En nuestros diseños, el consumo de potencia es de 84W y de 139W para los diseños Niagara2 y Niagara3 respectivamente [1].

Como se ha mostrado en la sección 0.3, el algoritmo $\text{MFA}_{\text{FU}*}$ colocará las unidades funcionales que componen el procesador, para minimizar los parámetros térmicos del chip. El área de éste se fija al principio del proceso de optimización y viene definido por la distribución original de los componenetes. Los resultados térmicos obtenidos por el optimizador serán comparados con los de los diseños originales mostrados en la Figura 7. Estas dos capas se disponen de modo que se pueda construir un sistema de 48 procesadores.

El trabajo experimental se basará en el análisis de la optimización térmica conseguida por el optimizador, y la reducción adicional de temperatura mediante la inclusión de los canales líquidos. Se han incluido además algunas



(a) (b)

Figura 7: Diseños originales. Niagara2 7a y Niagara3 7b.

modificaciones adicionales para la creación de dos dominos térmicos aislados mediante los canales de aire. Estos canales se colocan a 5400μm del lateral izquierdo del chip en las capas 1 y 3, y a la misma distancia del lateral derecho en las capas 2 y 4. Las fuentes de calor se colocarán en dichas regiones mientras que aquellas unidades funcionales con un menor consumo de potencia tenderán a colocarse en las vecinas aisladas.

Por otra parte, los diferentes algoritmos MFA se configuran con diferentes parámetros. Tanto el $\text{MFA}_{\text{FU}*}$ como MFA_{AC}, se configuran con una población de cien individuos y un número de generaciones igual al número de unidades funcionales, lo que evita que el algortimo se quede en un óptimo local. En ambos algoritmos, la probabilidad de cruce se ha fijado en 0.90 y la de mutación en 1/número de unidades funcionales, como se recomienda en [44].

Los algoritmos MFA_{TSV} and MFA_{LC}, se han configurado con una población máxima de cien individuos y un número máximo de generaciones de 250. La probabilidad de mutación depende del número de variables; en este caso particular es la inversa del número de puntos disponibles para insertar las TSVs o los canales líquidos. La probabilidad de cruce es de 0.90 como en el caso anterior.

0.5. Resultados

Todos los resultados que se mostrarán a continuación se han calculado mediante el modelo térmico presentado anteriormente en 0.2. Todos los diseños, a excepción del caso original, han sido obtenidos por los algoritmos $\text{MFA}_{\text{FU}*}$, MFA_{AC}, MFA_{TSV} and MFA_{LC}.

Además de las medidas de los objetivos fijados por la temperatura máxima (expresada en K) y la longitud de cableado (presentada en celdas, cada celda es de 300 μm), se ha incluido el gradiente térmico como una medida indirecta de la fiabilidad debida a la temperatura.

La Figura 8 muestra la distribución térmica del escenario original donde

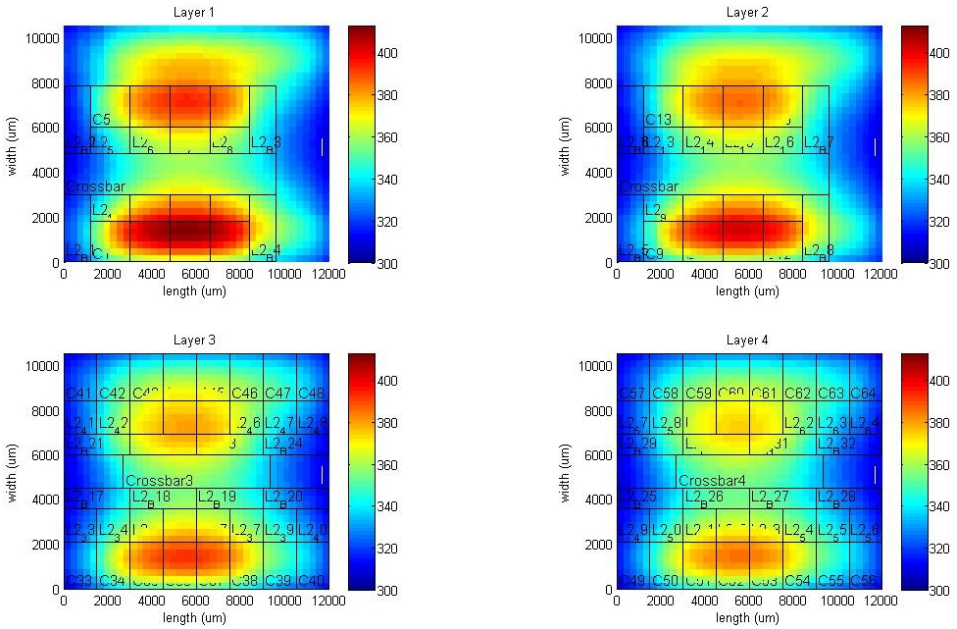


Figura 8: Mapas térmicos para los diseños originales de 48 procesadores

no se ha realizado ninguna optimización. Como puede verse aparecen regiones calientes, especialmente, en la primera y segunda capa ya que no son capaces de disipar el calor fácilmente.

0.5.1. Colocación de las unidades funcionales y las TSVs: $\mathbf{MFA_{FU*}}$ + $\mathbf{MFA_{TSV}}$

La Figura 9 muestra la solución alcanzada mediante el ($\mathrm{MFA_{FU*}}$+$\mathrm{MFA_{TSV}}$). Los cuadrados negros de las figuras muestran las posiciones en las que existe una TSV. Como puede verse en la Tabla 3, el optimizador ha conseguido una reducción de 38K en la temperatura máxima al compararlo con el escenario original.

La Tabla 3 compara las métricas para los diseños originales y aquellos obtenidos por los algoritmos $\mathrm{MFA_{FU*}}$+$\mathrm{MFA_{TSV}}$. Debe apuntarse que tanto los diseños del Niagara2 como los del Niagara3 están aislados lo que hace imposible la comunicación entre capas que ya están altamente optimizadas desde el punto de vista del cableado, y por lo tanto es imposible mejorar el cableado de los diseños originales. Del mismo modo, el cableado calculado por $\mathrm{MFA_{FU*}}$ (912 celdas), no se incluye en la Tabla por no ser éste realista.

Como puede verse, la propuesta de optimización es capaz de reducir la temperatura hasta en 37K. Aunque, como la Tabla 3 muestra, el rendimiento del sistema es peor por el cableado ya que en nuestras optimizaciones estamos conectando unidades funcionales colocadas en diferentes capas. Sin embargo,

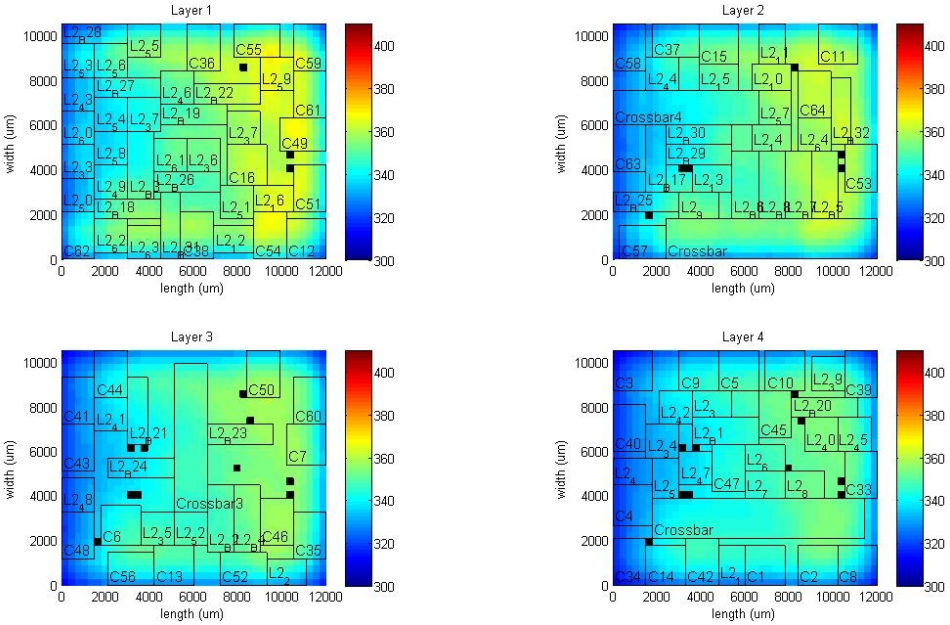


Figura 9: Mapas térmicos para el diseño de 48 procesadores optimizado.

Algoritmo	**Temp. Max.**	***Grad.***	**Cableado**
Baseline Stack (∅)	399	89	1012
$\text{MFA}_{\text{FU}*}$	362	43	-
... + MFA_{TSV}	362	43	1459

Tabla 3: Comparativa térmica y de cableado para la colocación de unidades funcionales y TSVs.

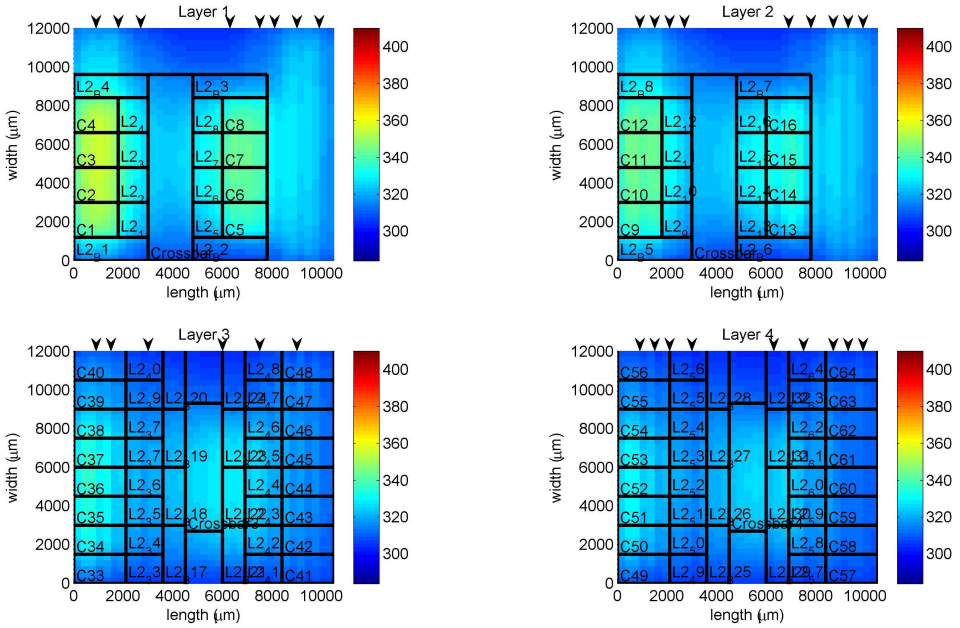


Figura 10: Mapas térmicos para el sistema original con 32 canales líquidos optimizados.

si se implementara el *wire bonding* como técnica de conexionado se podrían alcanzar las 2680 celdas, y muchas más si considerásemos el conexionado bidimensional. Esto prueba la conveniencia de usar las TSVs como la solución más prometedora para aprovechar las características de la integración 3D.

0.5.2. Liquid microchannel optimization: MFA_{LC}

Las altas temperaturas que se alcanzan en el diseño original pueden ser rebajadas mediante la inclusión de canales líquidos como un método eficaz para disipar las temperaturas en las capas internas. En este escenario se han introducido 32 canales líquidos. El mapa térmico con los 32 canales optimizados puede verse en la Figura 10. Como puede verse, las áreas calientes han desaparecido ya que los canales son capaces de enfriar el calor producido por los procesadores. Sin embargo, la colocación de los canales, como veremos a continuación también es importante. La comparativa entre colocar los canales de forma homogénea y de forma optimizada se muestra en la Tabla 4. La distribución homogénea (8 canales por cada capa) es capaz de reducir la temperatura, sin embargo, al hacerlo de forma optimizada no sólo se mejoran estos valores sino que además se mejora también el gradiente térmico.

El hecho de encontrar una distribución óptima para la colocación de los canales líquidos mejora no sólo las métricas térmicas sino que también reduce

Algoritmo LC	Temp. Max.	Grad.	Cableado
Baseline Stack ($\emptyset$)	399	89	1012
Homogéneo ($\emptyset$)	348	46	1012
MFA_{LC}	341	38	1012

Tabla 4: Distribuciones homogénea y optimizada de los canales líquidos en el diseño original.

Algoritmo LC	Temp. Max.	Grad.	Cableado
$\text{MFA}_{\text{FU*}}$ + $\emptyset$	362	43	1459
$\text{MFA}_{\text{FU*}}$ + colocación homogénea	339	26	1459
$\text{MFA}_{\text{FU*}}$ + MFA_{LC}	335	24	1459

Tabla 5: Colocación homogénea y optimizada de 32 canales líquidos en el diseño optimizado.

los costes de fabricación y la posibilidad de hacer más anchos los canales si éstos se colocan juntos, lo que también conllevaría una reducción de los costes energérticos de bombeo. La optimización de la colocación de los canales líquidos consigue muy buenos resultados, pero estas mejoras pueden acentuarse si además se usan otras técnicas como la colocación de las unidades funcionales.

De aquí en adelante, la colocación de los canales líquidos se hará de forma optimizada y se le aplicará en consecuencia al diseño optimizado anterior. Como ya se dijo, el optimizador de los canales líquidos no sólo evalúa aquellas zonas térmicas donde es más favorable colocar un μcanal, sino que también tiene en cuenta la colocación de las TSVs. En la Figura 11 se puede ver que una vez distribuidos los canales líquidos ya no aparecen zonas calientes. Al optimizar la colocación de las unidades funcionales y la colocación de los canales líquidos, se ha conseguido una reducción de 57K en la temperatura máxima, 33K en la temperatura media y 66K en el gradiente, respecto al diseño original.

La Tabla 5, muestra los datos del sistema optimizado al añadírsele 32 canales líquidos, tanto de forma homogénea como optimizada. El incluir los canales líquidos no afecta para nada la métrica del cableado, ya que no cambia el enrutado del mismo.

0.5.3. Aislamiento por canales de aire: MFA_{AC}

Una de las mayores contribuciones de esta tesis es la de añadir canales de aire para aislar los dominios térmicos en los que se ha dividido el chip. Este aislamiento hace que los procesos de enfriado sean más sencillos y eficientes, ya que los canales líquidos podrían colocarse en aquellas zonas donde la temperatura es más alta, reduciendo el número de canales líquidos para obtener

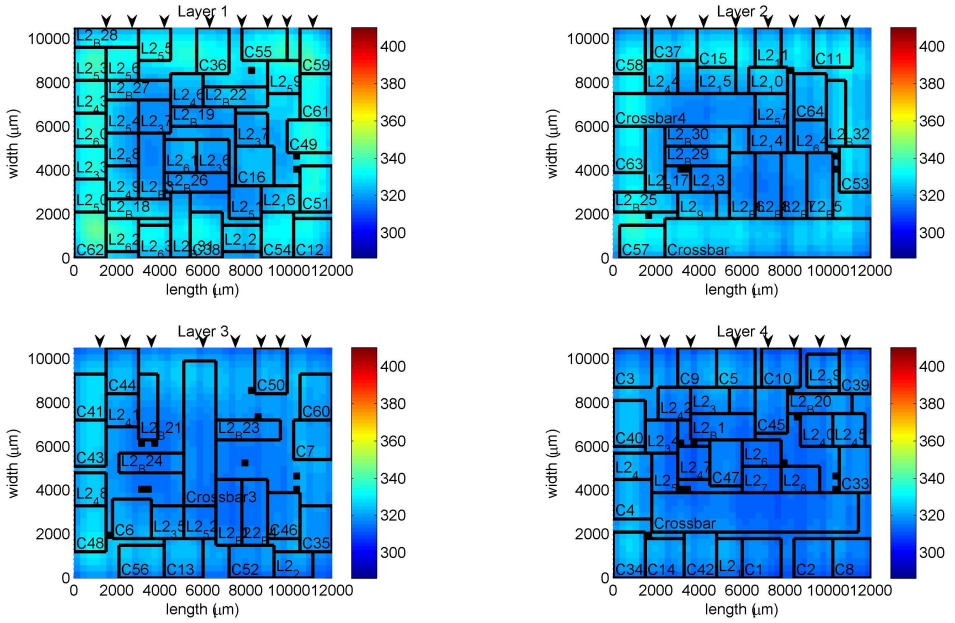


Figura 11: Mapas térmicos del diseño optimizado con 32 canales líquidos.

el mismo perfil térmico que sin ellos. Esta reducción en el número de canales implica una reducción en los costes de fabricación y en la energía asignada al sistema de bombeo.

Como se explicó anteriormente en 0.4, mediante el aislamiento se crean una región caliente y otra fría. La Figura 12 presenta la distribución inicial sin canales líquidos, donde el símbolo $*$ denota la existencia de un canal de aire. En la Figura 13 se muestra el mismo escenario pero después de haber colocado de forma optimizada 20 canales líquidos con nuestro algoritmo MFA_{LC}. Como puede verse en 12, el optimizador ha colocado las unidades funcionales con un consumo de potencia mayor en las regiones cálidas (zona izquierda de las capas 1 y 3 y zona derecha de las 2 y 4), mientras que las unidades con un menor consumo han ido a parar a las regiones vecinas. Una vez realizada la colocación de las unidades funcionales, los canales líquidos se colocan en el circuito 3D según se puede apreciar en la Figura 13. La mayor parte de los canales se colocan en las zonas calientes, sin embargo algunos se colocan en las zonas frias para paliar los efectos de la difusión vertical y conseguir un perfil térmico más homogéneo.

La Tabla 6, muestra los resultados al colocar las unidades funcionales en las zonas aisladas, así como los canales líquidos. Con este enfoque, al usar el MFA_{LC} en el diseño, se puede ahorrar la colocación de 12 canales líquidos para obterner los mismos resultados térmicos que se conseguían al colocar 32. Estos resultados se pueden ver en la figura 13 y las Tablas 5 y 6. Los

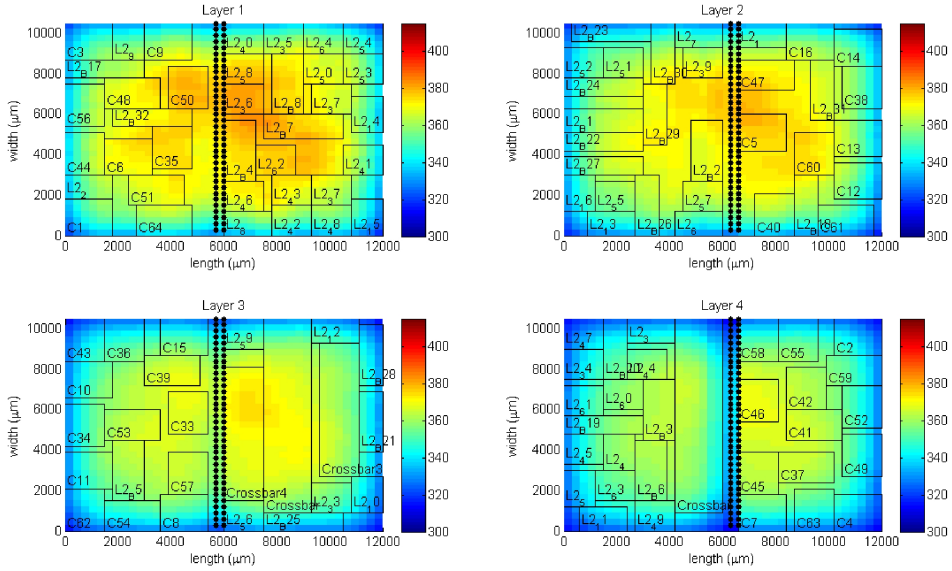


Figura 12: Distribución del diseño con aislamiento por canales de aire (MFA_{AC})

Optimizador	Modo Canales	# de canales	Temp.Max.	Grad.
MFA_{AC}	-	0	377	61
MFA_{FU*}	Homogéneo	32	335	24
MFA_{AC}	MFA_{LC}	20	335	27

Tabla 6: Comparativa entre el diseño optimizado con y sin aislamiento con canales de aire más la inclusión de canales líquidos (MFA_{AC} + MFA_{LC}).

resultados tienen un gran valor sobre el impacto térmico de las unidades funcionales más calientes, así como una mejora de los costes tecnológicos y de los costes de operación debido al sistema de bombeo. Dado que algunas unidades funcionales se separan mediante los canales de aire, existe un pequeño aumento en el cableado debido a la necesidad de encaminarlo por nuevas rutas. Este aumento se compensa con los beneficios térmicos y con el ahorro en el número de canales, que en este caso particular, de acuerdo con [93], supone un ahorro de 5W en los costes de operación.

0.6. Conclusiones

La integración 3D ha surgido como una solución a los límites físicos que están alcanzando los procesos de integración de dispositivos electrónicos. Sin embargo esta nueva forma de integración trae consigo algunos problemas relacionados con su fiabilidad, que está principalmente afectada por las al-

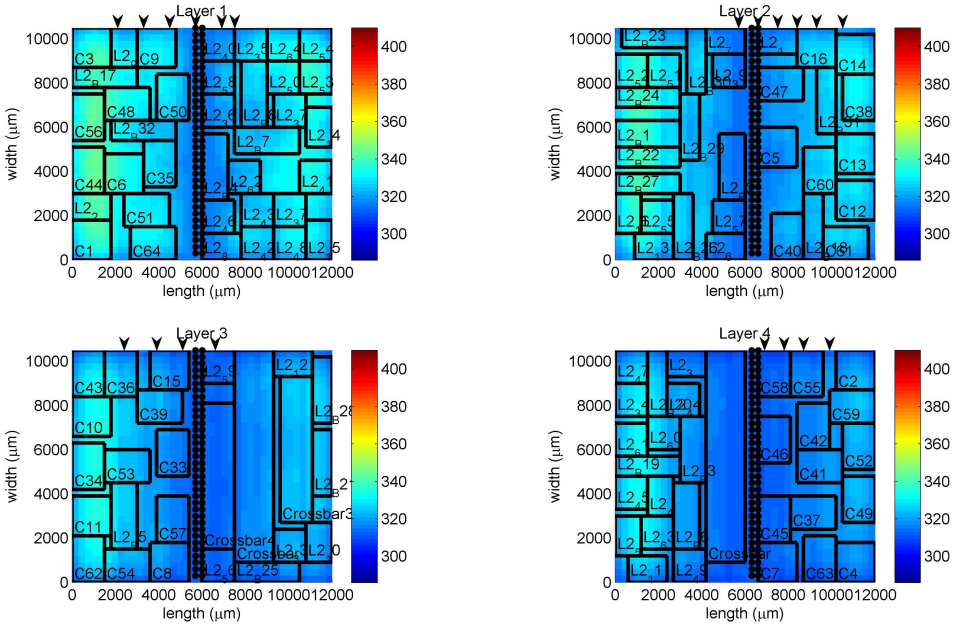


Figura 13: Mapa térmico del diseño resultante con canales de aire y 20 canales líquidos (MFA_{AC} + MFA_{LC})

tas temperaturas que se alcanzan dentro del volumen de diseño. Esta tesis propone soluciones a estos problemas en la fase de diseño.

Las capas internas de los dispositivos 3D no son capaces de disipar el calor que generan al ambiente, lo que afecta de forma negativa la vida util de los dispositivos. A pesar de que existen numerosos métodos tradicionales para sofocar el problema de la temperatura en sistemas en 2D, éstos no podrían aplicarse de forma eficiente en sistemas 3D, ya que éstos no son válidos por si mismos y hay que acompañarlos con novedosas técnicas específicamente diseñadas para esta tecnología.

Este trabajo presenta su estudio sobre un sistema multiprocesador real (Niagara) que se optimiza térmicamente mediante las herramientas de optimización propuestas. El optimizador presentado en la tesis es capaz de optimizar las posiciones de las unidades funcionales de acuerdo a su densidad de potencia, reduciendo el impacto térmico de las unidades calientes en el volumen de diseño.

Se ha presentado una nueva forma de representar las soluciones que permite el emplazamiento de las unidades funcionales a lo largo de todo el espacio de diseño y se asegura que la comunicación entre las diferentes capas sea posible mediante la inclusión de TSVs. Al combinar las técnicas de optimización de emplazamiento con la colocación de μcanales líquidos los

resultados térmicos se ven favorecidos.

Todo lo anteriormente expuesto se utiliza en una de las novedades de diseño propuestas en este trabajo, el diseño con dominios térmicos. Estas zonas térmicas se mapean en el volumen de diseño usando canales de aire en el interior de las capas activas. Con la existencia de estas zonas, se pueden crear regiones calientes y frías, de modo que será más fácil enfriar aquellas zonas caracterizadas por tener una mayor temperatura, obteniendo de este modo mejores resultados con menor coste.

0.7. Publicaciones

A continuación se listan las publicaciones científicas a las que el trabajo realizado ha dado lugar:

- D. Cuesta, J. L. Ayala, J. I. Hidalgo, D. Atienza, A. Acquaviva and E. Macii. "Adaptive Task Migration Policies for Thermal Control in MPSoCs". In Proc. *International Symposium on Very Large Scale Integration (ISVLSI)*. 2010, pp. 110-115.

- D. Cuesta, J.L. Risco and J.L. Ayala. "3D Thermal-Aware Floorplanner for Many-Core Single-Chip Systems". In Proc. *Latin American Test Workshop (LATW)*. 2011, pp. 1-6.

- D. Cuesta, J. L. Ayala, J. I. Hidalgo, M. Poncino, A. Acquaviva and Enrico Macii. "Thermal-aware floorplanning exploration for 3D multi-core architectures". In proc. *Great Lakes Symposium on VLSI (GLSVLSI)*. 2010 pp. 99-102.

- D. Cuesta, J.L. Risco and J.L. Ayala. "3D Thermal-Aware Floorplanner using a MILP Approximation". In Proc. *Design of Circuits and Integrated Systems (DCIS)*. 2010.

- D. Cuesta, J.L. Risco, J.L. Ayala and J.I. Hidalgo. "A combination of evolutionary algorithm and mathematical programming for the 3D thermal-aware floorplanning problem". In Proc. *Genetic and Evolutionary Computation Conference (GECCO)*. 2011, pp. 1731-1738.

- D. Cuesta, J.L. Risco-Martin and J.L. Ayala. "3D thermal-aware floorplanner using a MILP approximation". In Journal *Microprocessors and Microsystems*. 2012. Vol. 36, pp 344-354.

- D. Cuesta, J.L. Risco-Martin, J.L. Ayala and J.I. Hidalgo. "3D thermal-aware floorplanner using a MOEA approximation". In Journal *Integration, the VLSI Journal*. 2012, Vol. 46, pp. 10-21.

- J.L. Ayala, A. Sridhar and David Cuesta. "Thermal modeling and analysis of 3D multi-processor chips". In Journal *Integration -Amsterdam*. 2010. Vol. 43, pp. 1-15.

Chapter 1

Introduction

A journey of thousand miles starts with a first step

Confucio

From the discovery of the first semiconductor transistor by John Bardeen, William Shockley and Walter Brattain at Bell Labs in 1948, semiconductor industry has invested a lot of effort in achieving smaller and smaller devices, increasing the integration scale and reducing fabrication costs.

However this continuous shrinking process, that was stated by Gordon E. Moore in 1965 [111], "the number of transistors on integrated circuits will double approximately every two years", has derived in several issues that the designers have had to address, such as inner temperature rising or communication problems. This shrinking process have been mainly motivated by the seek of a better performance and by the experienced decrease in the fabrication budget of electronic devices. However, what Moore could not imagine was that this trend would find a ceiling in its climbing process due to the physical limits of "classical" fabrication techniques as Kurzweil, Sterling and Vinge have noted.

The continuous seek of a better performance in less space that has been adopted since 1980's have come upon many issues, not only in terms of performance or economic costs, which are the final goals of the technological designer, but also in terms of reliability and efficiency of the final designs. These problems have been addressed by the electronic industry using different designing and fabrication techniques, and proposing new architectural basis to cope with some of the found problems, as can be seen in Figure 1.1 where not only more transistors are integrated but also different architectures are used. Architectural solutions have to be seen as another way of fighting the war of performance/integration capacity..

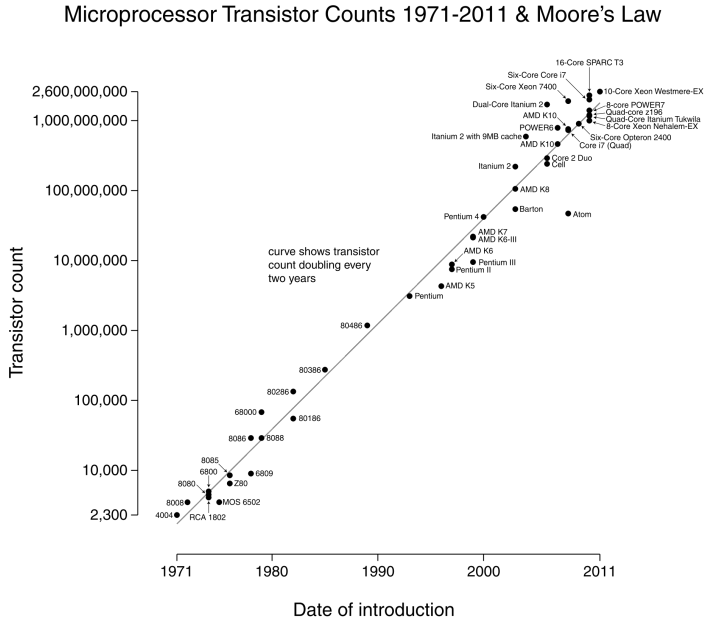


Figure 1.1: Evolution of the number of transistors in a single chip.

When the evolution trend started to be reduced in the first 2000s, designers alleviated the effect of loss in the performance proposing new techniques based on scaling physical components of the circuits in order to accelerate the processing units. These techniques are known as over clocking and consist of rising the clock cycle in order to make the processor quicker. However, these hardware techniques that came also with architectural innovations, rapidly found a limit, given by the loss of efficiency in terms of power consumption and the inherent temperature rise which leads to reliability issues and a reduce in the lifetime of the electronic devices.

The designing trend was starting to find its limits but higher performance and efficiency results were still required by the industry. At this step of the evolution of processing chips, designers proposed the new and revolutionary way of increasing performance reducing power consumption by including more than one processing unit in a single chip based on the results found in servers. That is how multi core systems were born.

The POWER4, developed by IBM, was a multi core microprocessor, with two cores on a single die and it was the first non-embedded multicore microprocessor. Multi core CPU designs require much less space in the chip package because some necessary hardware can be reused by many cores, and furthermore, these cores can use the same circuitry such as the L2 cache and the interface with the communication bus. Another benefit given by multi

core chips is also the allowance of higher performances at lower energy.

From the moment that fabrication industry realized that multi core chip was a big area of research with competitive and promising results, the investment of their efforts in multi core systems designs was highly risen, which was translated in a fast manufacture of chips with more and more cores in a single die as can be seen in Table 1.1.

As was previously stated, the integration of several cores in a chip, makes the thermal management of the multi core chip much more difficult than in single core packages. Despite the fact that Multi Processor System on Chips (MPSoC) achieve better performance with a smaller clock cycle, the global power density dissipated in the chip cell is higher, which affects the thermal behavior. When the quad-core processor was developed, Intel realized that the thermal problem should be carefully studied. Besides, processing power is not the only constraint on system performance, communication constraints have to be studied as well. As the number of cores is increased communication solutions have to be provided in order to comply with bandwidth requirements.

The inclusion of more and more cores in single die, makes the chip area wider, due to the physical limit that is being reached by the integration capability. As the chip is spread, the distance among functional units in it are risen, which directly impacts on the communication latency. Although the number of cores in a chip is scaled up in order to achieve better performance results, the communication system must provide a full and fast inter communication feature or it will become the new bottleneck for the whole system performance.

The actual problem of traditional integration is that the improvements in performance do not scale as the integration density does, mainly due to communication problems. The only possibility of finding limits in the integration of transistors in Integrated Circuits (IC), has led researches to propose new integration techniques capable of dealing with this problem, such as 3 Dimensions (3D) integration. As will be see later in more detail this technology is able to vertically integrate the disparate functional components of a System on Chip (SoC) as depicted in Figure 1.2, besides it allows the individual functional units of the system to be built in the process technology that optimizes their individual cost and performance, since each layer can be manufactured using different technologies. As the resultant performance of a SoC is the sum of its individual parts, the overall system performance and cost can be optimized using this technology.

However, as will be expounded in more detail in the next chapter, 3D integration technology comes by the hand of several issues regarding, over all, temperature and reliability problems. In order to address the thermal problems, due to the complexity of the design phase, automatic design tools take on vital importance. The design problem presented in this thesis lies in

Table 1.1: Evolution of the number of cores in a single die.

System	*Cores*	*Produced from:*	*Company*
POWER 4	2	2001	IBM
RAW	16	2003	MIT
POWER 5	2	2004	IBM
ULTRASPARC IV	2	2004	SUN
PA 8800	2	2004	HP
CELERON	2	2004	INTEL
PENTIUM D	2	2005	INTEL
AsAP 1	36	2005	UC Davis
ULTRASPARC T1 (NIAGARA)	4-8	2005	SUN
PA 8900	2	2005	HP
ATHLON X2	2	2005	AMD
VEGA 1	24	2005	Azul Systems
CORE DUO	2	2006	INTEL
OPTERON	2-16	2006	AMD
P8X32A Propeller	8	2006	PARALLAX
CORE 2 DUO	4	2006	INTEL
TURION 64 X2	2	2006	AMD
PENTIUM DUAL CORE	2	2006	INTEL
XEON	2-10	2006	INTEL
PLAY STATION 3	8	2006	SONY
CORE 2 QUAD	4	2007	INTEL
POWER 6	2	2007	IBM
TILE64	64	2007	TILERA
ULTRASPARC T2	4-8	2007	SUN
PHENOM	2-6	2007	AMD
Teraflops Research Chip	80	2007	INTEL
PHENOM II	2-6	2008	AMD
AsAP 2	167	2008	UC Davis
ATOM	2	2008	INTEL
CORE i3	2-4	2010	INTEL
POWER 7	4-8	2010	IBM
SPARC T3	8-16	2010	ORACLE
CORE i5	2-6	2011	INTEL
SPARC T4	8	2011	ORACLE
CORE i7	2-6	2011	INTEL
SEMPRON	2	2012	AMD
E64G401	64	2012	ADAPTEVA
PC205	273 DSP	2013	MINDSPEED

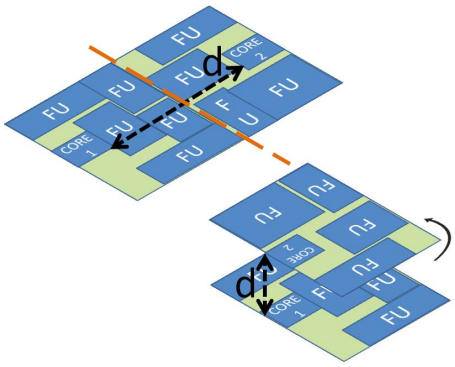


Figure 1.2: Communication latency reduction.

the possibility of creating feasible floorplans from the the thermal point of view.

3D integration offers many new possibilities in order to achieve better performance products with less power consumption apart from the develop of new heterogeneous architectures, as will be presented in Chapter 2. However, the industry has not completely accepted the challenge yet, since the manufacturing risks are already too high. Besides some fabrication processes such as wafer thinning, thinned wafer handling, alignment, bonding and vias creation, are not completely polished and many improvements have to be already done.

On the other hand, 3D testing is still giving its first steps, and further research have to be done in order to establish normalized testing methods and procedures. There exists a need in developing EDA tools for 3D design, capable of taking into account all the new features that 3D technology entails.

The main 3D designing problem, in which this PhD Thesis is focused, is the resulting high temperatures due to 3D stacking. As will be later shown in this work, two approaches can be taken to cope with the thermal problem. The first strategy is related to hardware techniques (Chapter 3) that change the structure of the package to reduce temperature. The second approach is based on CAD techniques (Chapter 4) that are applied in the designing phase to predict and alleviate the thermal impact in the chip.

Thermal-aware floorplanning has become a very important topic of research for 3D ICs. The basis of the method is to reduce thermal metrics such as maximum temperature, thermal gradients and mean temperature minimizing also the wire length in order to keep high performance metrics.

Thermal-aware floorplanners have been widely used in the design of more and more complicated circuits in traditional 2D integration which are incorporating more and more processors in a single chip. Some of these techniques have been exported to the 3D case, however they generally fail in

alleviating the high temperatures reached in inner layers, thus the necessity of developing new CAD techniques specially for the 3D integration case.

One of the main objectives of our proposal is to achieve feasible 3D designs from the thermal point of view, and to address the problem different design algorithms have been developed. The complete optimization flow will be able to alleviate the thermal stress in the chip by placing the functional units in strategic positions, as well as the TSVs and incorporating active cooling methods such as liquid μchannels and novel design techniques as thermal domains.

From the computational point of view, the exhibited problem of creating a thermal-aware floorplans by placing functional units is considered as an NP-Hard problem. The great amount of feasible floorplans that exist when the three dimensions are considered makes the problem even more difficult and traded-off solutions must be returned by the mathematical processes. As will be then shown, in order to attack the problem, different Multi-Objective Evolutionary Algorithms (MOEAs) have been implemented and used.

Despite there are no complete 3D design tools, there exist several thermal simulators for 3D stacks which are employed to compute the thermal behavior of the proposed designs, and specially those returned from the thermal-aware floorplanners. In our study, a developed thermal simulator will be used in order to check the feasibility of the floorplans generated by our optimization process.

In short, the integration of multiprocessor using 3D technology, is becoming a real important bet in the future of IC manufacturing. Since there exists a clear lack of thermal aware floorplanning methods for the 3D case, this thesis proposes the following goals:

- To show the architectural possibilities to cope with the thermal problem and the possibilities of 3D adaptation.
- To present different design strategies which take into account the thermal behavior of the 3D stack.
- To use a combination of hardware and design techniques that include active cooling system placement.
- To propose novel thermal aware floorplanning techniques based on a multi-objective optimization, able to propose feasible 3D scenarios while simultaneously optimizing different conflicting metrics, like performance and maximum temperature.
- To propose novel ways to design 3D stacks like the inclusion of air channels and thermal regions to optimize the cooling effectiveness of the proposed solutions.

- To use 3D thermal simulators to validate the results delivered by the design algorithm which can take into account all the cooling techniques accurately.

In the next the structure of the thesis is described.

In Chapter 2 a description of the 3D integration process is given. The main fabrication processes are presented with the difficulties and challenges that the integration technology has to face in order to make the 3D integration technology feasible.

Then in Chapter 3 several hardware techniques to alleviate the thermal problem in the traditional 2D integration and their possibilities and alternatives for the 3D scenario are described. In this chapter, the basis of the thermal domains proposal is explained.

Other kind of techniques, based on algorithms and design techniques are later presented in Chapter 4, where the optimization flow is perfectly described as well as state of the art thermal aware floorplanners. The thermal model used for compute the temperature distribution of the 3D stack is also detailed.

In Chapter 5, the scenarios used for the validation of the proposals of this thesis are depicted as well as the results obtained by the optimization algorithms.

Finally, in Chapter 6, the conclusions and the future lines of research are exposed. At the end of this chapter the publications related to this thesis are listed.

As an appendix to the thesis, a short description of the different optimization mechanisms used along the execution of this researching work, is attached.

Chapter 2

3D Integration

If the automobile had followed the same development cycle as the computer, a Rolls-Royce would today cost 100 dollars, get a million miles per gallon, and explode once a year, killing everyone inside.

Robert X. Cringely

Integration industry has focused their efforts in increasing the integration density of the chips, achieving better performance results and reducing final costs. However the shrinking process is arriving to its limit due to physical constraints. These series of improvements have achieved a notable increase in device switching speeds and hence the performance of the devices. Fundamentally, these improvements have been concentrated on transistors, while interconnect performance has been forgotten and fallen behind these new and faster transistors.

3D integration takes advantage of the geometric disposition of the functional units, using multiple vertical layers of transistors to improve performance, instead of the single layer of transistors that most modern integrated circuits have.

Power consumption and its efficiency is also considered as a very important topic of research in 2D integration. Interconnection mechanisms in modern designs can draw up to a third of the power utilization of the microprocessor mainly due to the big distances that electrical signals have to complete [38].

We have been talking about 3D integration, however, this term has to be taken carefully because modern "2D" chips are already built in three dimensions. They are built using various layers of interconnect material and the active layer is also grown in different layers and materials. 3D integration term is adopted when the whole system is built using multiple layers of silicon

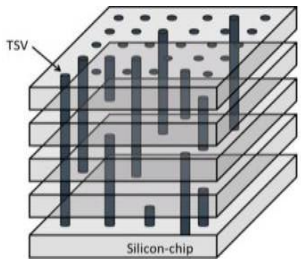


Figure 2.1: TSVs in a 3D chip.

substrate, each one containing transistors and metal connections that run on top of each substrate as in conventional 2D chips. These different layers are arranged in a stacked configuration, one on top of the other. At this point is when the obvious question rises, how can we communicate different layers? This question is resolved by using Through Silicon Vias (TSV). These vias are tunneled directly through layers in a vertical direction as depicted in Figure 2.1.

One of the most significant benefits that 3D integration provides is the reduction in wire length as was previously illustrated in Figure 1.2. This reduction is automatically translated in a lower power dissipation since less electronic communication elements are needed, as well as the decrease in wire length.

On the other hand, a lower communication latency among blocks can reduce the cycle time, increasing frequency and hence, chip performance. When different layers are stacked the integration density concept drastically changes because it has to be now considered as the number of transistor per volume unit. Since more "traditional" chips are included in the same package, fabrication cost are also reduced due to the reduction of I/O pins which makes the final package simpler and hence cheaper. This fact puts into discussion the validity of the Rent's rule as Das et Al. published in [41]. The rule that has been used in 2D integration to determine and analyzing the circuitry, determining communication complexity, power consumption and fabrication costs has to be now adapted to the 3D integration case.

Opposite to all these improvements that 3D integration offers, there is also an advantage that 2D traditional integration does not allow, the integration of heterogeneous technologies as depicted in Figure 2.2, where specific optimized active layers are stacked one over the other.

If this integration had to be done in traditional 2D ICs, it would have required a penalty in the performance of any type of circuitry, because every element can be optimized using different integration technologies. On the other hand, three dimensional integration offers the possibility of using different technologies for each layer, since every layer can be treated as independent, allowing the optimization of every layer for performance, without

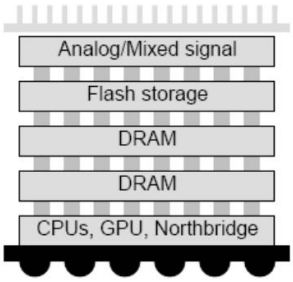


Figure 2.2: heterogeneous 3D chip.

compromising the performance of the other ones. An SoC is the sum of its individual parts and if different technologies and functionalities such as analog, logic, memory and I/O are individually optimized, then the overall system performance and cost is optimized.

In sort, the result of integrating different device layers comes with several benefits:

1. Wire length can be reduced due to the features of 3D geometry. Since the distance among functional units is decreased, communication latency is lower which results in a higher performance of the system.

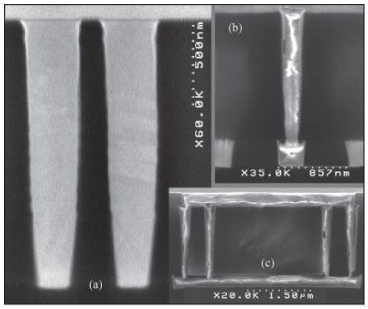
2. The power dissipated by the whole stack is also reduced because the clock cycle can be decreased as the power dedicated to communication systems.

3. Since each layer is independently manufactured, 3D circuits allow the integration of heterogeneous systems, which is optimum for dedicated systems.

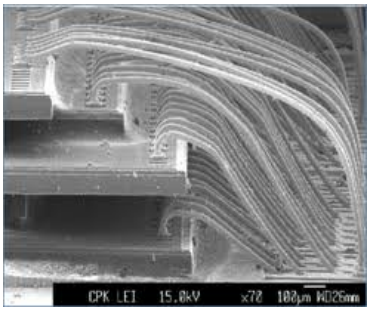
4. The vertical dimension adds a higher order of connectivity and offers new design possibilities.

5. The stacked structure complicates attempts to reverse engineer the circuitry. Sensitive circuits may also be divided among the layers in such a way as to obscure the function of each layer.

Although 3D integration exhibits great improvements it has also its own drawbacks which mainly comes with the temperature issues. Despite the global power consumption is reduced, certain parts of the 3D stack could increase their power density because of the stack of different layers, plus the difficulty of refrigerating those inner hot regions. Without careful attention early in the design and simulation of the chip, the resulting thermals could reach unacceptable levels, affecting device reliability and requiring expensive cooling solutions. At this level, the use of Electronic Device Automation (EDA) tools, that could take into account the temperature levels of the



(a) (a) Cross section of 1.6um high via, (b) cleaved SEM image of isolated via, (c) cleaved SEM image of 175nm diameter vias [77].



(b) 3D wire bonding

Figure 2.3: Inter layer communication in 3D integration.

chip, becomes essential. This problem will be later addressed and will be the goal of this PhD thesis.

2.1. Manufacturing 3D Chips

As has been exposed, the only way to make 3D integration work is to connect different layers among them. The most important techniques to allow inter layer communication are the fabrication of TSVs (Figure 2.3a) or manufacturing wire bonding (Figure 2.3b).

Wire bonding consists of route wires among layers in order to allow inter layer communication, however it requires great in and out-plane area, which is inconsistent with the objective of maximizing component integration density, thus TSVs, as the one depicted in Figure 2.3a, appear to increase their importance in 3D integration, as the only "effective" way of communicating layers. Conventional wire bonding techniques produce inductive losses that would reduce the speed of data exchange, impacting directly in the device performance. On the other hand, TSVs are able to eliminate the drawbacks that wire bonding exhibit, besides many other benefits are offered. TSVs allow to reduce the wire length between the die, thus power consumption caused by long horizontal wiring is reduced, as is the space used for communication elements such as buffers which are no longer required. Adding TSVs to the design reduce electrical parasitics in the circuit which is translated into an increased switching speed. Moreover, TSV accommodates much higher input/output density than wire bonding, which consumes much more space.

Although using TSVs is much more efficient than wire bonding, integrating TSVs in a 3D stack is more difficult and comes with some challenges for the 3D manufacturing industry. The most constraining challenge is the

implementation cost. According to "Applied Materials" [105], the value of integrating TSVs in a 3D design could mean a 30 % cost increase over wire bonding. Therefore, designing phase has become a very important challenge in order to make TSV integration a profitable mechanism to cope with the inter layer communication.

TSVs manufacturing has to be integrated in the whole fabrication process in order to provide the greatest cost-effectiveness relation. The traditional steps that are taken to process wafers has to be combined in order to fabricate TSVs. There are two different approaches to manufacture TSVs, "via first" or "via last". Depending on which is taken the process sequence differs. In either approach, the TSV stack undergoes bonding, thinning, wafer processing on bonded/thinned wafers, and subsequent de-bonding. As will be later shown, once the wafers are bonded the temperature to ensure the robustness of the junction can not exceed 470 K, which introduces a new constraint in the manufacturing process, the thermal budget. The TSV process can be done in three main steps, via creation, via lining and via filling.

Initially, vias were created using laser ablation, however, this process is not effective if the number of integrated vias is increased since it rises concerns regarding damage and defects. Opposite, the well proven in traditional semiconductor manufacturing process known as ion etch, has become the most important technique to create the holes for TSVs. If the fabrication process is "via first", the vias are etched during the front-end-of-line processing (same steps in which the transistors are made), on the other hand, if the "via last" approach is taken, vias are created during or after back-end-of-line processing (same step in which communications are formed) from the front-side of a full-thickness wafer or backside of a thinned wafer. Which technique is used and when depends mainly on the device that is fabricated. While the "via first" process is preferred by the logic suppliers, the "via last" approach is followed by image sensors and stacked DRAM manufacturers. In either approach the goal in to etch as fast as possible the wafer in order to obtain a good profile control [74].

To avoid shorting to the silicon, the etched vias must be lined with an insulating layer of oxide before being filled with metal. If a copper fill is used, a second metal barrier liner is also needed. The process is well known in traditional integration. Chemical Vapor Deposition (CVD) is the most extended method to create the thin film of isolating material as depicted in Figure 2.4. The used isolating material must have good breakdown voltage and obviously very low leakage currents. The deposition of the electric barrier is one of the most challenging and expensive processes in the TSV fabrication flow. To this end titanium or tantalum are used if the via is then filled with copper. For the deposition of these material another well known technique is used, Physical Vapor Deposition (PVD).

Once the hole for the via is prepared it has to be filled. The most com-

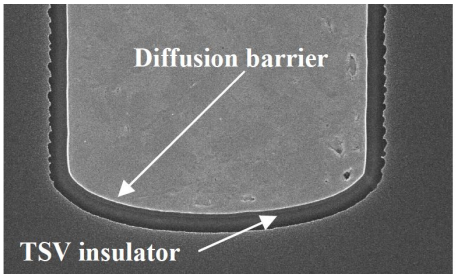


Figure 2.4: Cross-section of TSV bottom, showing Bosch-etch striations and fully-filled via.[77].

mon filling material due to its physical and electric characteristics is copper, however other materials such as tungsten can be also used in order to obtain better aspect ratios [160].

Therefore as has been shown, manufacturing 3D integrated devices is not as easy as it was in 2D. There are two main methods to manufacture 3D electronic devices, "bottom up" wafer fabrication and the "bottom down". The "bottom up" method, consists of building silicon layers sequentially on top of each other. The first layer is formed and transistor devices are fabricated, followed by the deposition of the second layer and the subsequent fabrication of its devices. However, this particular method exhibits some issues regarding the reliability in subsequent layers that are deposited on top of the already built ones. The main advantage is that the size of the inter layer vias can be scaled down.

The "tow down" technique manufactures each layer separately and afterwards bonds them together. This has come to be the most extended method for 3D fabrication due to the advantages that it presents. The main reason is that no changes have to be applied in the already developed manufacturing industry, since each layer is independent from each other. The fact of not changing the already existing manufacturing techniques has a direct impact on the quality of the wafer and hence, its reliability. Another very important advantage is that this method, opposite to the previous one, offers the possibility of integrating heterogeneous silicon layers, each one optimized for separate process functions. However this method comes also with some drawbacks. One notable issue is that the size of the inter layer vias is not expected to scale at the same rate as the transistor devices. Even in the best case, vias cannot decrease below one micrometer in width, because of inter-inter layer bonding alignment tolerances, which is the most important challenge of the method.

When the "top down" is applied, three different bonding processes, as depicted in Figure 2.5, can be applied:

- Wafer to wafer: Electronic components are built on two or more semi-

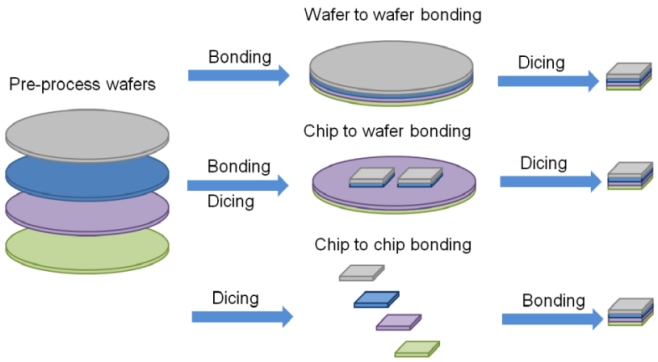


Figure 2.5: "Top down" manufacturing processes [52].

conductor wafers, which are then aligned, bonded, and diced into 3D ICs, despite aligning entire wafers is more difficult it exhibits great industrial throughput [103]. Each wafer may be thinned before bonding. Vertical connections are either built into the wafers before bonding or else created in the stack after bonding. The main disadvantage of this method is that if one of the wafers in the stack is defective the entire 3D stack will fail.

- Chip to chip: Electronic components are built on multiple dice, which are then aligned and bonded. Thinning and TSV creation may be done before or after bonding. One advantage of chip-to-chip is that each component die can be tested first, so that one bad die does not ruin an entire stack. Moreover, each die in the 3D IC can be binned beforehand, so that they can be mixed and matched to optimize power consumption and performance.
- Chip to Wafer: Electronic components are built on two semiconductor wafers. One wafer is diced; the singular dice are aligned and bonded onto die sites of the second wafer. As in the wafer-on-wafer method, thinning and TSV creation are performed either before or after bonding. Additional dice may be added to the stacks before dicing.

In Table 2.1, a summary of the main features of the different integration technologies is displayed.

2.2. Challenges of the 3D Integration

It has been seen through the presentation of the different manufacturing methods, the difficulties that each process entailed. In the following the main difficulties and challenges that the 3D manufacturing industries have to cope with is presented.

Table 2.1: Comparing 3D Integration Technologies.

Stacking Method	*Wafer to wafer*	*Chip to Wafer*	*Chip to chip*
Production yield	Low	High	high
Production Throughput	High	Depends on alignment accuracy	Low
Main Applications	DRAM	Heterogeneous systems	Packaging

2.2.1. Stacking Methods

It has been stated that "top down" process is the most popular consisting of stacking independent layers, but how can this process be done? How are the wafers or the dice stacked? There are two main methods in order to create the 3D stack. Wafers and dice can be bonded face to face (Figure 2.6 (a)), or face to back (Figure 2.6 (b)), however it must be noted that in an entire 3D chip, different stacking method can be used in the different layers (Figure 2.6 (c)).

If two layers are stacked using face to face method it will make the tops of two layers stack facing each other with their interconnect layers exposed and connected by vias. As the interconnections are faced, final stack will result in the minimum possible wire length and the minimum via size, which is translated into an increment in the interconnect densities. This is the obvious assembly for only two layers systems, or the last two layers of a stack, otherwise the distance with the next layer would be higher and the benefits of using this method would not be so clear.

On the contrary, face to back, each layer is stacked on top of another, all having the same orientation. The distance between layers is larger and the vias must be larger as they have to go through the silicon substrate of each layer, forming a direct vertical interconnection. The wafer layers can be thinned, achieving better electrical characteristics and control. For three dimensional stacking with more than two layers, face to back bonding is the only viable approach in unlocking the true promise and full benefits of the third dimension.

2.2.2. Alignment and Bonding

If stacking different layers in order to create 3D devices were easy, the manufacturing industry would have already moved to just 3D integration. However these methods entail many processes that have to be carefully done.

Alignment represents a very important issue from the point of view of TSV integration which is translated into an impact on the density of vertical connections. The alignment tolerance depends on the stacking processes

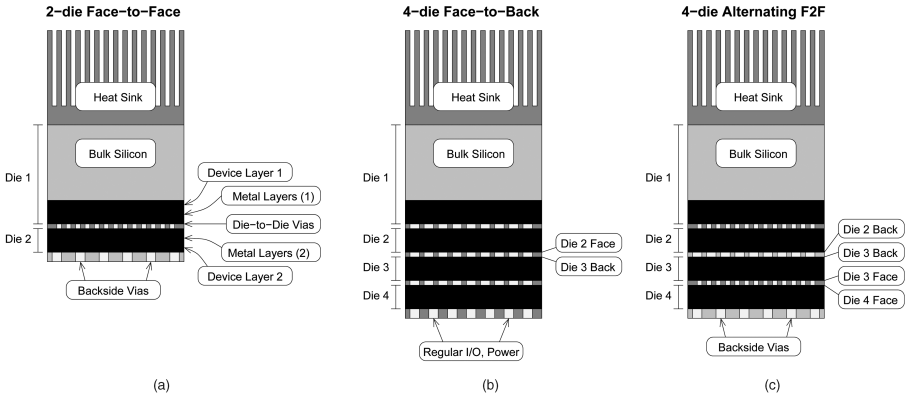


Figure 2.6: (a) Face to face (b) Face to back (c) Combination [124].

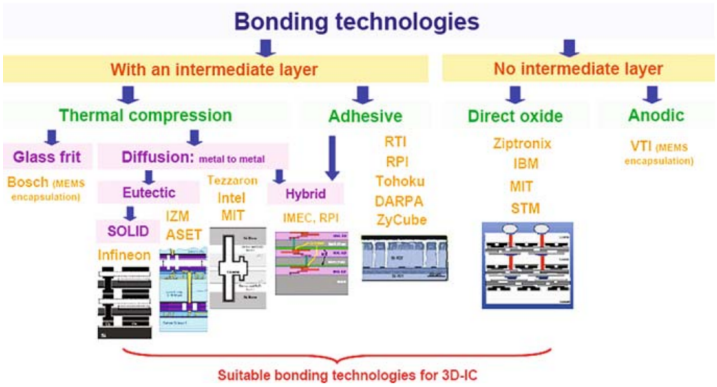


Figure 2.7: Bonding Strategies([45].

shown before. The best results are obtained for chip to chip alignment achieving tolerances of about 200nm [162], on the other hand the accuracy in the wafer to wafer is about 1μm or even smaller [125].

On the other hand, bonding strategies open a full branch of research as can be seen in Figure 2.7. The bonding process consists of a mechanical process that applies force in order to press two different wafers or dice into direct contact, while the materials are heated under defined conditions of temperature and pressure so the chemical and mechanical bonding processes can take place. In short, bonding process consists of creating the most accurate and uniform mechanical distribution in order to make the different layers stack perfectly. There are four main Wafer-Bonding Techniques for 3D.

The first one is **Silicon Direct Bonding**. This method is based on intermolecular interactions including van der Waals forces, hydrogen bonds and strong covalent bonds. The initial procedure of direct bonding was characterized by a high process temperature. Hence, the aim consists of achieving a stable and hermetic direct bond at a temperature below 720 K. Therefore

processes for wafer surface activation, i.e. plasma treatment or chemical-mechanical polishing (CMP), are taken into consideration and are being researched. The upper limit of 720 K bases on the limitations of back-end CMOS processing and the beginning of interactions between the applied materials. The drawback of the silicon direct bonding method is that the surface must be flat and atomically smooth. It is generally accepted that the surface roughness must be approximately 1 nm or less.

The next process is **Adhesive Bonding**. As its own name shows, it uses adhesive materials to bond the different materials. Despite there are many adhesive materials that could join the materials the most commonly used are the SU-8 and Benzocyclobuten (BCB), which are specialized for Micro Electro Mechanics (MEM) or electronic component production. Adhesive bonding has the advantage of relatively low bonding temperature as well as the absence of electric voltage and current. Based on the fact that the wafers are not in direct contact, this procedure enables the use of different substrates. However with this method is very difficult to obtain an accurate intermediate layer in terms of dimension control. Besides there exist a possibility of corrosion due to the mechanical nature of the adhesive materials if the process is not carried out in optimum conditions.

The third more extended bonding process is the **Copper-to-Copper Diffusion Bonding**. In order for diffusion reactions to proceed, the Cu surfaces must be in intimate contact. Copper to copper diffusion bonding must be done at 670-720 K and for 20-40 minutes. The most important advantage that this method presents is that it allows electrical connections between the layers.

The last process consists of **Eutectic Bonding**. It is based on the ability of silicon (Si) to alloy with numerous metals and form a eutectic system. The most established eutectic formations are Si with gold (Au) or with aluminum (Al).Eutectic bonds are a special class of metal bonding in which the metal melts when it reaches the eutectic temperature. The alloy composition corresponds to a triple point on the phase diagram as is shown in Figure 2.8, in which the solid-to-liquid transition is spontaneous and does not involve the coexistence of both liquid and solid phases simultaneously. In the Figure the Au-Sn phase diagram is presented. The direct melting path is depicted in dashed lines. The second path is drawn in dots and corresponds to the diffusion taken through several two-phase regions until the eutectic composition and solidification.

2.2.3. Wafer Thinning

Wafer thinning or backgrinding, is the process in which the thickness of the wafer is reduced to allow for stacking and high density packaging of integrated circuits. Typical silicon wafers used for integration purposes have diameters of 20 or 30 cm. The thickness of these wafers is about 750 μm

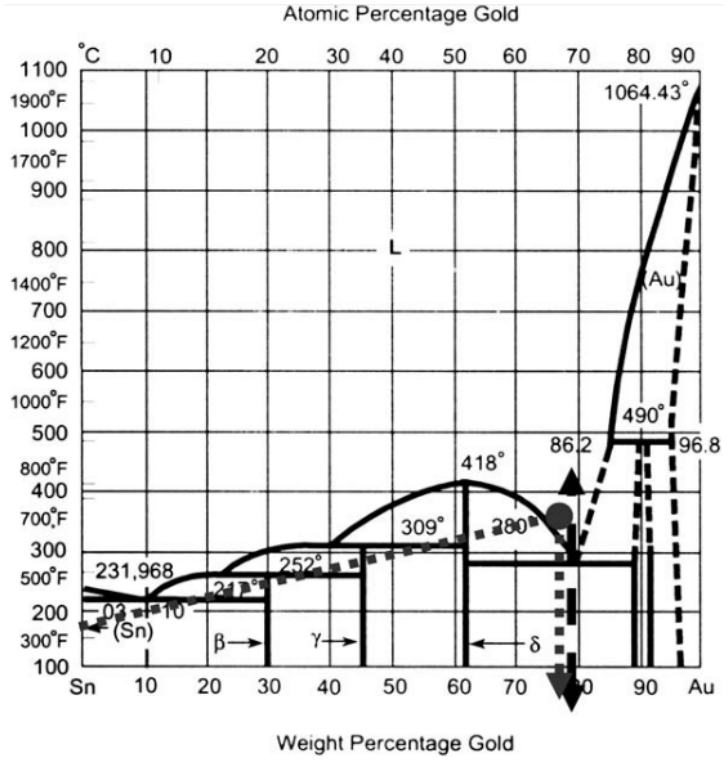


Figure 2.8: The Au-Sn phase diagram.([145].

in order to ensure mechanical robustness and to avoid warping during high temperature manufacturing processes. Since 3D integration uses back to face stacking methods, the distance between one layer and the other is constraint by the back thickness of the wafer, and this non processed thickness is not small enough to allow 3D integration being profitable. With the current processes wafer can be thinned up to 20-50 μm in order to ensure the robustness of the processed wafers when handling them.

The thinning process is typically carried out using etching mechanisms. Since the required uniformity of the thinned wafer has to be extreme (no more than 1-2 μm) in order to allow later bonding, the etching process has to be stoped using different approaches such as an ion implanted layer, a graded SiGe layer, or a buried oxide layer [160].

2.2.4. Yield and Test

The final 3D IC fabrication issue that semiconductor designers and manufacturers face is related to the yield and test. One previous assumption that has to be done is that the whole 3D integrated device can be assembled at the wafer scale, before testing, or at the tested die level. From the definition of 3D it may be certain that yield is improved since it allows more

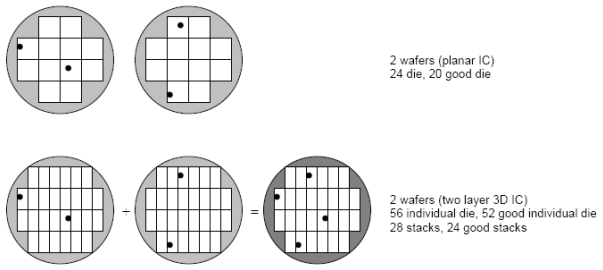


Figure 2.9: Wafer to wafer yield example.

integration, however this certainty must be carefully studied because it will depend not only on the integration kind but also in the steps taken for its manufacture.

Wafer to wafer bonding of large dice has a detrimental effect on yield, since as more wafers are stacked together, the more likely the whole chip stack is to be ruined by one bad layer as was previously stated. However, if the die size is shrunk the effect on the yield could be very different. If two dice that are about half the size of the original integrated circuit are stacked, yield will actually improve. Specifically, breaking a planar die into multiple smaller pieces will increase the yield of the smaller dice, since more candidate dice can fit in a given wafer. Because there are more candidate dice per wafer, the number of defects, will affect a smaller percentage of the overall number of candidate dice. This is depicted in Figure 2.9.

On the other hand, when chip to chip stacking method is used, the employed elements are of known good quality which means that the manufacturing yield is controlled before the attachment, however, the manufacturing throughput is reduced because each die is processed individually instead of the whole wafer. Even considering this fact, the obtained yield using die to die method is higher than in the wafer to wafer bonding case. In the particular case of the example depicted in Figure 2.9, die to die bonding would have created 26 out of 28 good chips instead of the 24 good chips produced using the wafer to wafer.

In the field of 3D integrated systems testing there is a wide opened line of research. In order to achieve a robust technology each layer of the stack should be able to be tested independently. However this is not as easy as it was in the traditional case where the wafer presented probe contact points and a full functionality which is no longer true in the 3D case, and many techniques pre-bonding [22] or post-bonding, [115] have been presented. Test issues are considered one of the major challenges of 3D integration [94].

2.3. Manufactured 3D Chips.

As it has been presented 3D manufacture is a hard task that is being addressed by some companies and institutions. In these few lines four examples of already manufactured 3D chips will be briefly presented.

In 2004, Intel presented a 3D version of the **Pentium 4 CPU** [13]. The chip was split in two dice, and the stacking was chosen to be face to face, which allowed a bigger density of vias. This stacking technique was chosen because Intel has one chief goal for 3D integration: reducing the length of metal interconnects.

Intel designers had to manually placed the functional units in each die, trying to minimize power density without losing performance. Since the Pentium 4 processor core has mainly logic blocks, a minimal 3D arrangement of two stacked die was sufficient to prove most of the benefits of 3D IC without compromising the resulting design due to power density issues. Logic elements switch and consume power at a much higher rate than memory circuitry. This presented a problem since existing thermal issues could be exasperated due to hot spots in the design. To remain within the thermal limit of the original design, very active power regions could not be placed on top of each other, as this would increase power density and temperature. As a prove of the future of the 3D technology, the new design exhibited a 15 % performance improvement (due to eliminated pipeline stages) and 15 % power saving (due to eliminated repeaters and reduced wiring) compared to the 2D Pentium 4 [13].

The **Teraflops Research Chip** [80], was introduced in 2007 by Intel. The project consists of an experimental 80-core design with stacked memory. To improve upon that, Intel designers implemented a TSV-based memory bus achieving great communication bandwidth at a very low power cost.

In ISSCC 2012, two **3D-IC-based multi-core** designs using GlobalFoundries [54] were presented. The chip consists of a 64 custom core implementation with two-logic-die stack.

By its side, the EPFL is developing a modular 3D stacked multi-processor architecture [58]. The platform is composed of identical stacked dies connected together by TSVs. Each die features four 32-bit processors and associated memory modules, interconnected by a 3D NoC, capable of routing packets in the vertical direction.

2.4. Reliability of 3D ICs and the Thermal Problem

Three dimensional stacked integration presents a unique and novel solution to the interconnection problem that is becoming more and more important in the 2D traditional integration. 3D stacking permits simultaneously improve performance while reducing power and wire length connections. Ho-

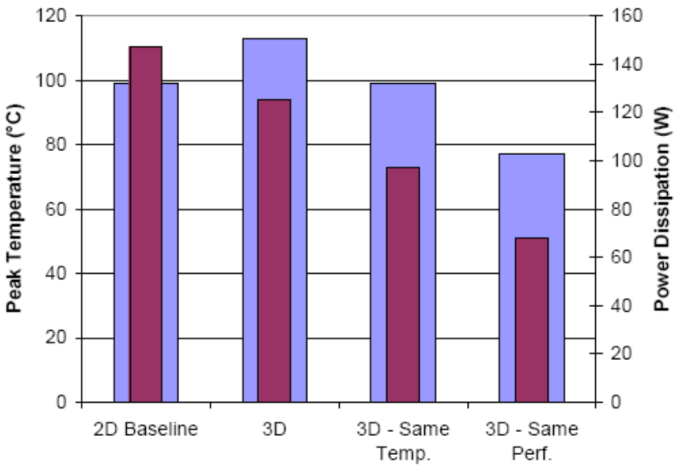


Figure 2.10: Pentium thermal and power results.[13]

wever, as has been previously stated, 3D integration has also some drawbacks. One of the most important is the temperature control of the 3D stacked devices.

The Pentium IV experiences driven by Intel [13], proved that although the power consumption in the whole chip is reduced, the reached temperature is risen as is depicted in Figure 2.10.

Reliability has become one of the main goals of the manufacturing industry since it directly impacts on their profits. As can be seen in Figure 2.11, there can be many failures that can affect the chip, most of them motivated by the device working temperature. Therefore the importance of being capable of controlling the temperatures inside the chip is crucial, since, as it will be later shown, temperature has a very high impact in electronic devices reliability. Temperature impacts on the reliability of the package but there are also other parameters such as humidity, mechanical or electrical stresses that also affect the device lifetime.

Considering all the factors that can influence package reliability, thermal stress has the most significant impact. The temperature behavior in ICs is a complex function that depends on many variables such as part construction, electrical operating conditions, air flow around the package, placement functional units, etc. As may be obvious the problem is even bigger in 3D integration where dissipation to the environment is more difficult. As a result the thermal characteristics of integrated circuits are a major concern to both users and producers of ICs. The increase in temperature due to a junction temperature (T_J) rise of the components can adversely affect the long-term reliability of the product.

Reliability issues have been widely and long studied. Manufacturers have based their reliability test on the trend depicted in Figure 2.12, where three

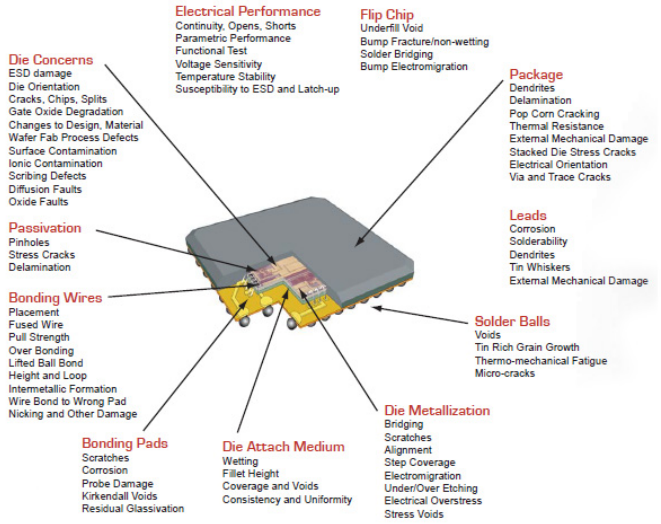


Figure 2.11: Failure Mechanisms [91].

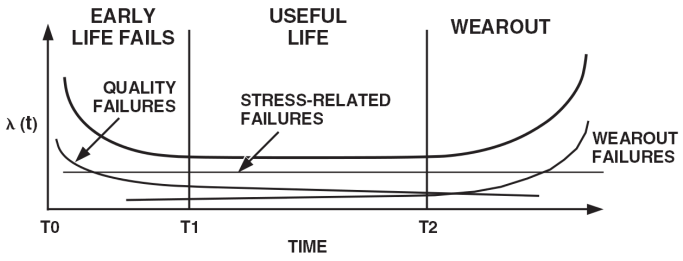


Figure 2.12: Bath tube curve reliability.

different regions are shown. The first one is assigned to the infant mortality which is usually restricted to the first months of circuit life. The failures that devices have in this period are usually due to manufacturing processes and are found at board level. Fabrication industries try to reduce as much as possible this stage ($T_0 \approx T_1$) since the fabrication guarantee supports the buyer of the product.

The next stage is the useful life of the device. The failures found in this period are characterized by a constant rate (λ). This failure rate is the number of devices that will be expected to fail in a given period of time which can be translated into a Mean Time to Failure. This period of time is very long ($T_2 >> T_1$) for integrated circuits, it can last for even decades. After this period, the life of electronic devices enter in the wear out stage, where failure mechanisms as oxide wear out or electromigration appear. Temperature, together with other environmental factors can reduce T_2 accelerating the failure of ICs.

The relationship between temperature and the reliability of the product is very well established and it is mathematically modeled by the Arrhenius equation 2.1:

$$\lambda = Ae^{-\frac{E_a}{k_B T}} \tag{2.1}$$

where λ is the failure rate, A is a characteristic constant of the electronic device, E_a is the activation energy of the failure mechanism typically around 1 eV, k_B is the Boltzman's constant and T is the temperature.

Stress Migration is one of the most important thermal driven failure mechanism that often occurs in the metalization of integrated circuits [69]. It is caused because the metal interconnect lines can go open circuit as a result of stress which can lead to open circuit or unacceptable resistance increase which cause the whole circuit to fail. The stress migration is generated by a thermal mismatch between the metal interconnects and the passivation film or the interlayer insulating film. Since there exists a stress in the board metal atoms migrate to alleviate the forces. However this movement can result in the creation of an open circuit or, if the communication line is not completely opened, in a reduction of the effective width of the metal line which can produce later electromigration mechanisms.

Industry has invested a lot of effort and budget in creating reliability tests in order to be able to predict the durability of their manufactured devices. One of the most common practices is accelerate the failure process increasing the temperature or even creating temperature cycles which also affect chip's lifetime.

Temperature cycle tests refer to tests used to evaluate the device resistance when exposed to high and low temperatures, and also the resistance when exposed to temperature changes between these two temperature extremes. These tests are very important because it simulates chip thermal gradients and changes in the operating conditions. Life related to these temperature differences has been modeled by Coffin-Manson low cycle fatigue equation formulated in 1953, and is expressed as follows.

$$\text{LifeTime} = A(\Delta T)^m \tag{2.2}$$

where A and m are constants and ΔT is the temperature difference.

Despite temperature is a very important factor in reliability it has been observed and reported, that in the temperature working range of 220-425K most of the failures are not due to high or low temperature but could be related to other thermal phenomena such as thermal gradients, temperature cycles or rate of change of temperature [92].

Chapter 3

Architectural Solutions for the 3D Thermal Issue

I'm all for simplicity. If it's very complicated I can't understand it.

Seymour Cray

As was previously shown in chapter 2, the computation industry is constantly growing, integrating more and more transistors in a single chip, achieving great computation performance. In its growth, designers must look for new and more aggressive ways to cool microprocessors, because traditional techniques, used for lower integration scales are not longer useful to alleviate the thermal problem, hence temperature becomes a big problem when we have to deal with 3D integration.

In this chapter we will describe the most important techniques to mitigate the thermal problem. These techniques have been applied to 2D designs and now, some of them will be extended to 3D structures. However, some new techniques have been developed exclusively for 3D stacks.

Classical techniques for reducing temperature in 2D devices are no longer useful for 3D structures because inner layers, where the thermal problems are more pronounced, cannot take advantage of them. Some of these techniques include external elements such as fans or liquid circulation circuits, that could only interact with the top layer of a 3D stack.

In this chapter we will present useful hardware tools, to decrease temperature throughout the chip, and some new techniques that adapt traditional 2D techniques to novel 3D integration systems.

3.1. State of the Art

In order to extend a chip lifetime, or to ensure a good performance, the operating temperature of a chip must be kept as low as possible, because high temperatures can negatively impact on transistors performance as has been previously stated in chapter 2.

The increased integration scale has led to higher power densities, which is automatically translated into higher temperatures. Hence, 2D designers have faced this problem with external elements that can contribute to heat dissipation.

These external gadgets can be mainly classified as passive or active elements.

3.1.1. Passive Elements

Passive elements are widely spread as a relative good quality/cost technique. The solution is based on heat interchange between the hot surface of the chip, and the colder temperature of the air.

The strength of the passive elements relies on the simplicity of the solution. Heat sinks are just made by a conductive metal, such as aluminum or copper. Their durability is impressive because heat sinks have no mechanical pieces, and over all, heat sinks have a low cost. Another advantage for computer users is that passive elements do not produce annoying noise that can disturb users comfort.

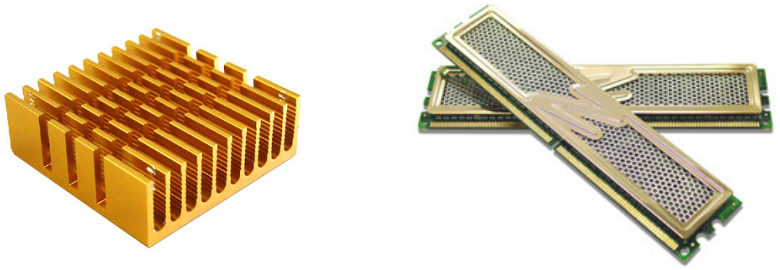
However, thermal dissipation through heat sinks has its limits, because they are not capable of spread heat to the air fast and effectively. Heat sinks were used without any auxiliary system until Pentium processors appeared in the market. Aluminum or copper heat sinks were not even able to dissipate first Pentium heat on their own.

In the market, not only heat sinks for microprocessors can be found(Figure 3.1a, but also for mother boards or memories as the ones showed in Figure 3.1b or low performance Graphics Processing Units (GPU).

Heat sinks are used to optimize heat interchange between air and chip package. Fourier's law establishes that heat flow is directly proportional to contact surface and thermal conductivity. For example, if we have a $1cm^2$ chip and a $1000cm^2$ heat sink we can compare how effective is heat spread with and without the heat sink, comparing the product of thermal conductivity of the materials and contact surface.

Air thermal conductivity is $0.02W/(m^2K)$ and copper thermal conductivity is $380W/(m^2K)$. When the package is directly in contact with the air the product would be:

$$0.01m^2 \text{ x } 0.02W/(m^2K) = 0.0002W/K.$$



(a) Commercial Microprocessor Heat sink

(b) Commercial Memories Heat sinks

Figure 3.1: Microprocessor and memories heatsinks.

If we add the heat sink the situation will be completely different, the interface package-heat sink would have a product given by:

$$0.01m^2x380W/(m^2\Delta K) = 3.80W/K$$

and the interface heat sink-air will be $10m^2$ x $0.02W/(m^2K) = 0.2W/K$. As we can see, the bottleneck is always the interface with the air.

Despite heatsinks can be seen as a very simple element many centers are still researching how this passive element can be optimized. The design of the heatsink can be done focusing in different aspects:

- The final purpose of the hardware to be cooled. [148] designed a special heatsink for a laser, or [25], that proposed a heatsink fabrication process for microwave and millimeter-wave devices.
- The materials that compose the heatsink. [53] proposed a special heatsink for optoelectronic applications made of cooper foam.
- Additional features. Heatsinks can be used not only to dissipate heat from hot surface, but also can perform other functions such as electromagnetic shields ([153]) or antennas ([33] or [3]).

Furthermore, heatsinks have a wide usage in space hardware, because in the vacuum radiation is the only way to dissipate heat from the electronic devices [15]. These heatsinks are designed to optimize the radiation in the infrared spectrum.

3.1.1.1. Other Passive Elements

Combining the principle of heat interchange that occurs in heat sinks with the air, and the properties of natural convection given in liquid fluids, a new passive element is born, heat pipes.

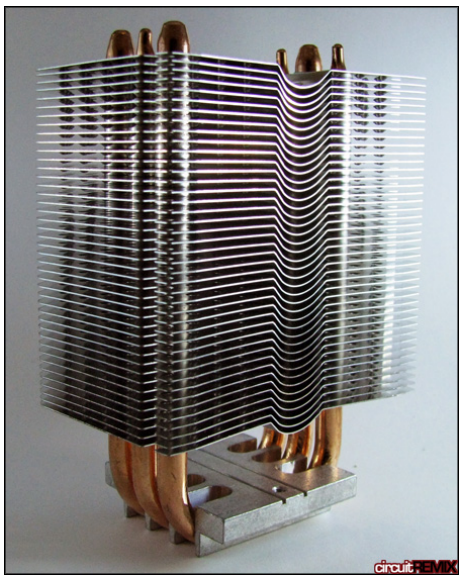


Figure 3.2: Commercial Heat Pipe used in OCZ Vendetta (circuitremix.com).

A heat pipe, as the one shown in Figure 3.2, is a thermal machine that works based on natural convection. When a fluid warms up, its density decreases and vice versa. If the base of the heat pipe is in contact with the hot surface of the chip, the liquid contained in it will go up to the upper part of the heat pipe, and the still cold liquid will descend, causing a natural circulation, which means a decrease in the temperature of the chip.

The circulation of fluid, due to natural convection, is not enough to cool down the chip. Besides, a very big structure is needed to make the system effective.

The observed results establish that passive elements are not able to dispose of enough heat to prevent microprocessor from damage, which leads us to a next step in cooling techniques, the usage of active elements.

3.1.2. Active Elements

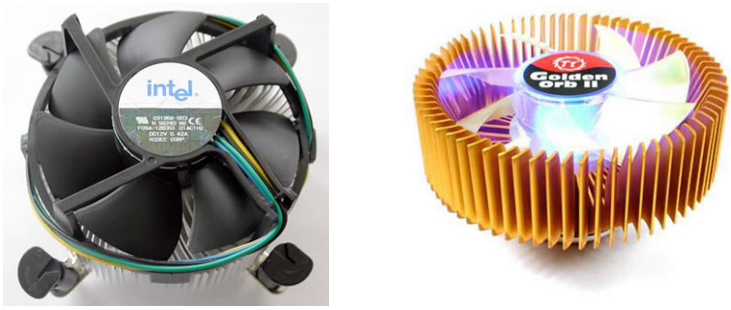
Active elements consist of different hardware components that consume power to work. These additional hardware elements are usually added to a passive cooling system. In the next paragraphs we will present the most useful and extended active cooling systems.

3.1.2.1. Fan Cooling

Passive heat sinks heat up the air surrounding the microprocessor and the heat sink. When the surrounding air is warm the cooling property of the heat sink is reduced drastically, because heat interchange depends on the

temperature difference between both environments, chip and ambient air. The surrounding air is moved by natural convection, but this effect is not fast enough to maintain a good cooling response. Including a mechanism to force fresh air into the heat sink, makes the convection much more efficient, helping with the cooling effect.

Among all the active cooling techniques, fans are the most commonly used. There are a lot of different fan models in the market, with several shapes and sizes. Fans are usually attached to heat sinks fabricated in aluminum or copper, two cheap materials with good thermal conduction. Two different models can be seen in Figure 3.3.



(a) CPU INTEL Fan for 775 slot (b) Copper Thermalake fan

Figure 3.3: Heat sink with a fan.

Fans are moved by a DC engine. This principle makes easy to switch the fan on if a threshold temperature is reached, or even tune its speed by controlling the current to the engine as is shown in [104].

Fans are the main system for many operating architectures and computation infrastructures. This fact makes this system the cornerstone of the cooling process hence failure must be avoided at all cost. That is why, some authors as [109] or [96], have developed techniques in order to monitor the state of the fans and prevent their failure. However, as it will be shown in the followong, these techniques are not suitable for 3D technology.

3.1.2.2. Liquid Cooling

A much more efficient way to cool a microprocessor down is the usage of a flowing liquid. Some liquids, for example water, have a higher specific heat constant, and a better thermal conductivity than air. These two parameters define the efficiency of heat transfer from the hot package.

The key of the flowing system is the pumping. Liquid cooling systems usually use water because of its low prize and its good thermal parameters and its harmlessness. However, some systems include oil in their circuits, because oil has better thermal properties, but on the other hand its viscosity

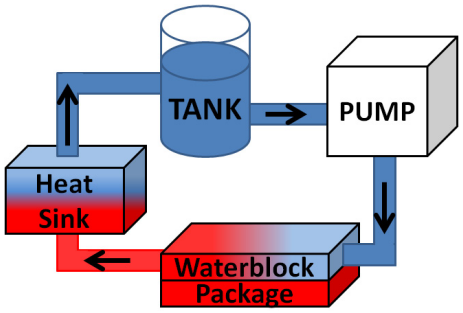


Figure 3.4: Water cooling system.

is higher so bigger circulation pipes are needed. In Figure 3.4 it can be seen a scheme explaining how the system works.

Liquid pump, moves cold water through the circuit. When the water goes into the waterblock that is in contact with the chip, heat transfer occurs between the hot surface of the package and the cold water that is flowing through the pipes inside the waterblock.

In this phenomenon heat from the package is dissipated very fast and is absorbed by the coolant. The heat now stored in the coolant is then dissipated by the heat sink. The liquid circuits counts with a tank where the liquid is stored.

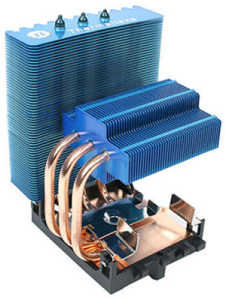
Liquid cooling system is a very effective way to dissipate heat from hot areas of the chip, and it can be done very fast. Liquid systems are also very varied and they can be adapted to space requirements and thermal needs. Four different systems can be seen in Figure 3.5.

The external system just presented is the most commonly used when liquid cooling is required, but there are other methods that are used in specific applications or are being investigated, such as liquid cooling by immersion or inner chip cooling using liquid microchannels.

Examples of immersion methods are very hard to find. Some researchers are still publishing studies with new liquids [4] or immersion for special architectures as [90] or [66]. However this technique is not used for commercial purposes and is usually restricted to personal computer designing convections and very overclocked supercomputers. In these systems, as can be seen in Figure 3.6, the whole system is immersed in a liquid.

This liquid must fulfill several constraints because it cannot conduct electricity, so its electric conductivity must be close to zero, and of course, the liquid cannot be corrosive. Another important property to guarantee a good cooling is to ensure that the thermal conductivity of the liquid must be as high as possible.

On the other hand, liquid microchannels consist of the fabrication of channels inside the chip. With this method not only the top surface of the



(a) Oil cooling system (Thermaltake)

(b) Liquid cooling system with an extern radiator

(c) Asus GPU with oil cooling

(d) Coomercial liquid circuit (Thermaltake)

Figure 3.5: Liquid cooling systems.

chip can be cooled, but also inner areas where the hot spots are created. As will be explained in more detail in Section 3.1.3.4, microchannels are one of the most important line of investigation for cooling 3D chips.

3.1.2.3. Thermoelectric Cooling Systems

Thermoelectric cooling is a commercialized solid state cooling method, presented as an alternative to the ones described above. There are also several systems based on vapor compression cooling and cryogenic cooling, but these techniques require large logistic footprints and are typically not suitable to use as precision spot cooling of microelectronic devices.

Thermoelectric cooling systems are based on Peltier effect . In 1834 Jean Peltier established that if a DC current passes through a P-N semiconductor there exists a temperature differential at the junctions.

In the movement from P to N type semiconductor, electrons leap to higher energy state which causes heat absorption phenomenon, which makes this side of the thermoelectric module to become cold. When electrons move from the n-type semiconductor back to the p-type semiconductor their energy drops down to a lower state and therefore, electrons release heat (hot side).

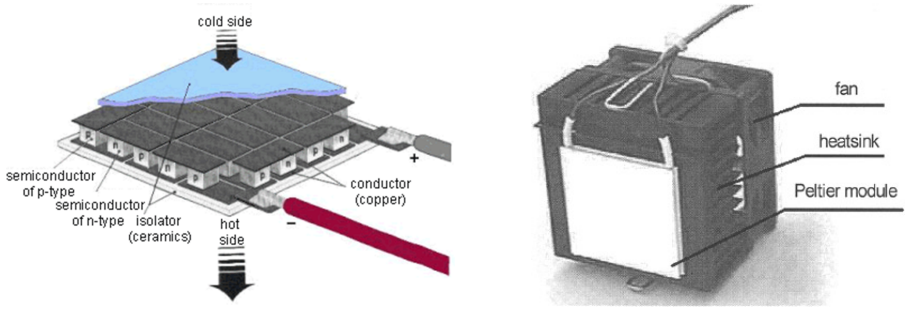
Figure 3.6: Immersion cooling system (www.armari.com).

Electrical current controls the thermoelectric effect, the more current, the bigger the temperature difference is. Heat current $Q_{Peltier}$, is proportional to the electric current, I, and the Peltier coefficient, Π as shown in equation 3.1.

$$Q_{Peltier} = \Pi I \tag{3.1}$$

For more information about this effect, the reader is referred to [81]. Peltier effect has been improved to use semiconductors as a heat pump, moving heat from one surface to the other one [130]. Peltier cells can be usually found in industrial environments but some manufacturers have developed solutions for chip cooling as the illustrated in Figure 3.7b

Typically, a commercial Peltier cell contains thousands of P-N junctions mounted in series, to multiply the temperature effect. One of the most important things of Peltier cooling technique is that materials are semiconductors, so they can be shrunk using integration methods.



(a) Peltier cell (b) Commercial Peltier system

Figure 3.7: Peltier cooling systems (http://ixbtlabs.com).

Peltier cells are used in combination with other cooling systems, active or passive, such as air fans or liquid circuits, because the hot side of the cell

must dissipate heat.

3.1.3. 3D Cooling Systems

As can be easily understood by the reader, these previously presented techniques have some drawbacks when they have to be applied to 3 Dimensions integration chips. In 2D techniques we can cool down the top surface of the chip, which was enough for a 2D chip, but in 3D thermal problems appear especially in inner layers, where these methods cannot address the heat dissipation. A summary of the previous presented techniques for 2D designs and their 3D adaptation capabilities is displayed in Table 3.1.

In this section, the most important techniques, found in the literature, will be presented, showing their strengths. Some of these techniques are then applied in our experimental work.

3.1.3.1. Placement of Functional Units

Distribution of the functional units inside the chip can have a very strong impact on the thermal profile exhibited by the package. Higher temperatures will be reached in the areas surrounding functional units whose power consumption is higher.

In order to keep a good thermal profile in the entire 3D stack and to reduce reliability problems as explained in Chapter 2, an homogeneous distribution of the functional units in the design volume must be achieved. The power dissipated by a unit also impacts on the temperature of their neighbors, increasing their temperature and hence, the thermal metrics that negatively affect the reliability of the chip.

Many authors have focused their work in placement techniques for 3D integration, because the thermal impact of the floorplanning on the thermal distribution of real microprocessor-based systems can be very high. This impact is analyzed in [156], where the placement of components for Alpha and Pentium Pro is evaluated.

In this way, thermal aware floorplanning has become an important area of research. Some initial works on thermal aware floorplanning as [26] propose a combinatorial optimization algorithm to model the problem. However, the simplification of the considered floorplan and the lack of a real experimental framework, motivated the further research on the area.

Thermal placement for standard cell Application Specific Integrated Circuits (ASICs) is a well researched area in the VLSI CAD community, where we can find works as [21]. Some authors, as [131] or [73], research this topic at the microarchitectural level, where it is shown that significant peak temperature reduction can be achieved by managing lateral heat spreading through floorplanning.

In our experiments, this topic will have a very important impact on the

Table 3.1: 2D cooling system summary

Technique	Pros	Cons	3D adaptation
Heatsink	Low cost Very well studied Used in vacuum environments	Must be adapted to space constraints Cannot dissipate high power densities	Can be used as an auxiliary method but cannot address the thermal problem in inner layers
Heatpipes	Good thermal dissipation No noise	The installation requires big spaces	As heatsinks, heatpipes are not a solution for 3D cooling
Fans	Low cost High efficiency Controllable speed	Noisy Exposed to failures. Need maintenance Heatsinks or heatpipes are needed to optimize the cooling	As the previous methods could be considered as an auxiliary technique
Liquid Circuits	Very good thermal response	Power consumption by the pumping Need maintenance	Using the same principle, μchannels are a good solution to the 3D thermal problem.
Thermoelectric Cells	Good thermal dissipation Controllable by DC current	Another heat dissipation system is needed Power consumption	Can be adapted to μcells to transport heat from inner layers

achieved results. Hot functional units must be placed as close as possible to the edge of the chip and far away from each other, which means, spread throughout the stack. If we are able to optimize the positions of all our functional units their power dissipation will have a lower impact on the global thermal behavior and an homogeneous thermal profile can be attained.

A 3D design scenario composed by forty eight cores in a chip can show the importance of functional units distribution. In Figure 3.8a cores are placed together. The heat dissipated by each core directly impacts on the others and the maximum temperature reached by the whole chip is higher than in Figure 3.8b, where the location of the cores is optimized in order to decrease the impact of each core dissipation. As can be seen in the Figure this optimization has decreased the maximum temperature eliminating the existence of hot spots.

3.1.3.2. Thermal Through Silicon Vias (TTSV)

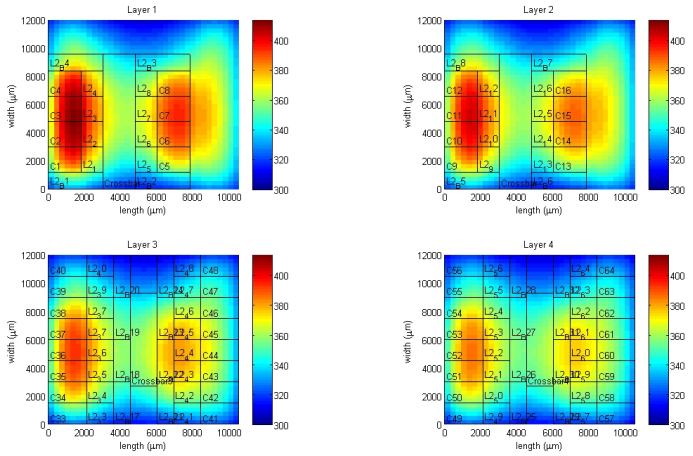
In a 3D structure electronic components such as semiconductor chips with different active IC devices may be provided in a multilayer stacked structure. In this case, inter-layer communication problems appear.

Traditionally wire bonding has been used to establish electrical interconnects between chips in different layers. It consists of route wires among layers in order to allow inter layer communication. Some authors such as [18], [82] or [150], have studied the effect and the advantages of this kind of integration. However wire bonding presents some deficiencies. The most important one is that the technology requires great in and out-plane area, which is inconsistent with the objective of maximizing component integration density.

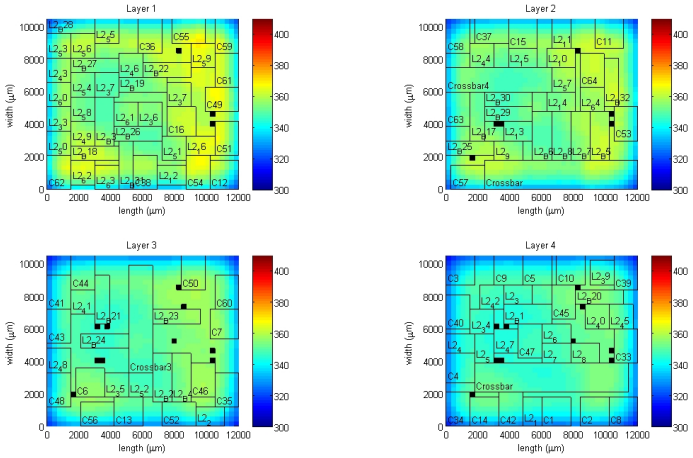
Some real images (3.9a, 3.9b)have been provided by the École Polytechnique Fédérale de Lausanne (EPFL). They show how wire bonding is performed in a real chip. It seems to be obvious that many "effective" area of the stack is wasted because of external wiring. Besides, usually wire bonding requires an extra "interposer" layer between the chips, to route the communication.

Due to the inconveniences that wire bonding has ([47]), a new technique has been developed in order to electrically connect two different components that are placed in different layers of the 3D stack, Through Silicon Vias (TSVs).

TSVs replace edge wiring by creating vertical connections through the body of the chip, hence, the resulting package does not need to be enlarged. TSVs do not need extra layers as wire bonding does, so chips whose inter layer communication are allowed by TSVs are flatter than an edge-wired 3D package. This TSV technique is sometimes also referred to as TSS (Through-Silicon Stacking or Thru-Silicon Stacking).



(a) Cores together



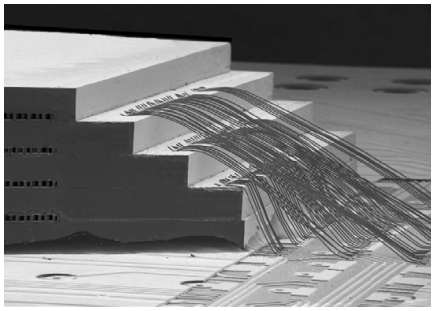
(b) Separated cores

Figure 3.8: Functional unit placement example.

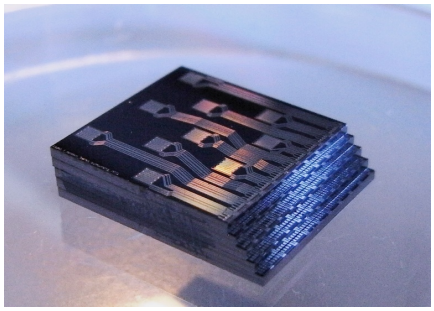
But TSVs can not only allow communication among components in different layers of the chip, but also decrease temperature in some areas inside the package. With this idea Thermal Through Silicon Vias (TTSVs) are born.

TSVs are pure metal objects that can spread heat inside the chip. Placing TSVs in strategic places in the stack can decrease the temperature in hot areas, transporting heat to colder regions.

Many publications such as [30], [64] or [9] bet for TTSVs solutions. A schema of the basis of the principle of heat transfer through TSVs is presented in Figure 3.10. As was previously said, TTSVs, due to their metal composition, can easily dissipate heat from hottest areas, transporting it to the heat sink, or other colder places. The effect of adding TTSVs connecting hot areas to the heat sink, improves thermal metrics in the chip like peak



(a) Microscopic View of Wire Bonding



(b) Real 3D Chip

Figure 3.9: 3D Wire Bonding.

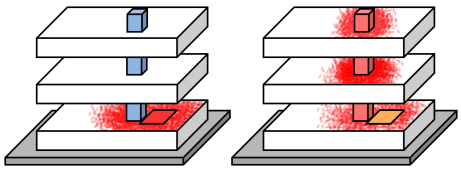


Figure 3.10: TTSV effect.

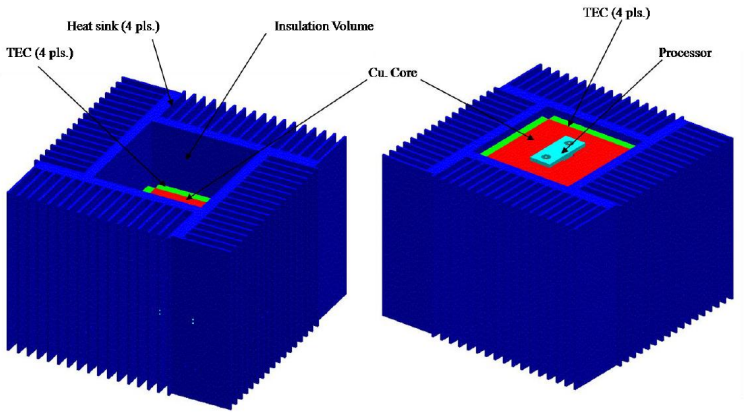
temperatures or heat gradients, however presents some drawbacks like the area overhead due to the routing of these TTSVs. Other approaches build an homogeneous thermal grid composed by TTSVs to spread heat throughout the 3D package minimizing thermal gradients within the same layer.

3.1.3.3. Thermoelectric cooling in 3D ICs

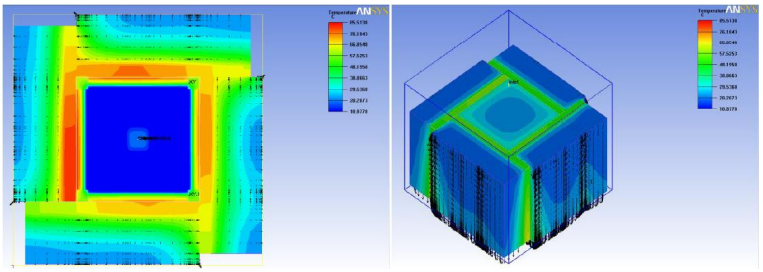
Thermoelectric effect was presented in 3.1.2.3. With this physical effect we can decrease the temperature of the chip using Peltier cells composed by semiconductors.

There are several proposed methods in the literature. Some of them, as the one presented in the PhD Thesis [59]. The main contribution of the thesis consists of creating a cold cage around the 3D IC made by Peltier cells and heat sinks. Results of this idea can be seen in Figure 3.11.

Heat sinks with the Peltier cells create an isolation region in which the 3D IC is placed. With the cold atmosphere created inside the cage the 3D stack can be cooled much easier. However, this solution does not face the problem of hot spots in inner layers. To this end, [122], integrates in the chip several micro Peltier cells. These cells force heat circulation towards the heat sink. As it is known, temperature differences in the faces of the Peltier cell can be controlled by the current, enabling a dynamic usage of the integrated cooling system, depending on instantaneous conditions.



(a) Heat sink offset configuration



(b) Peltier cage result model

Figure 3.11: Peltier cage.

3.1.3.4. Liquid Microchannels (μchannels)

In 3D ICs inter layer liquid cooling seems to be a good solution to address the high temperatures reached in 3D stacks. As was previously seen, liquids have better thermal properties than air, which enables fast heat removal from the hot regions of the chip.

However, forced convective liquid cooling has some drawbacks. As can be seen in [144] one problem is the high pressure encountered in the microchannels due to the usage of external pumping systems. External pumping systems are unsuitable for MPSoCs, because of their power consumption and large size.

A typical microchannel consists of very small parallel channels whose width is on the order of 50-500 microns. Current technology to create such microchannels may use photo-chemical etching or other machining techniques. Electro static discharge machining (EDM) could be another approach, but it would be very costly. High costs and difficult methods for creating the microchannels in the stack are one of the major problems of liquid cooling.

Another problem is the integration of the pumping system, that is why, some authors, as [137] are focusing their research in microscale pumping tech-

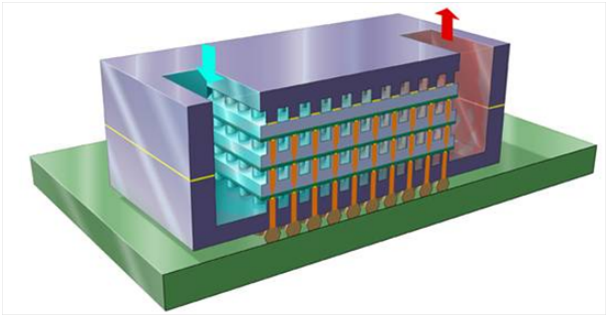


Figure 3.12: IBM conceptual 3D microchanneled chip.

nologies, which offer significant advantages because they are much smaller than conventional pumps and they can be integrated into the microchannels, reducing also fabrication costs and power consumption. But still micropumps exhibit several challenges to achieve both high flow rates and high pressure drops, because individual Micro Electro Mechanics (MEMs) could not create sufficient pressure head and flow rate to make the liquid flow, so multiple micropumps should be distributed through the length of the channels.

Studying the response and the design of micropumps is out of the scope of this Ph. D. thesis, but a very good overview of the different micropumping systems can be seen in [137].

Microchannels are a very good option to cope with the thermal problem without impacting on other design techniques like TSVs placement. As can be seen in the conceptual 3D IC by IBM (Figure 3.12), microchannels can perfectly coexist with TSVs.

3.1.4. Limitations of the State of the Art

In the next paragraphs the main cooling techniques that can be used in 3D integration technology will be summarized (Table 3.2), presenting their deficiencies and contributing with some ideas that will be then developed.

Heat sinks are the most common used option for cooling a chip, however, they have only effect in the top layer, which is useful in 2D technology, but it is not enough for 3D integration, where the most important temperature problems are found in inner layers. Heat sinks, will be used in 3D technology as a back-up method, as are being used with current 2D chips.

In consonance with heat sinks, fans can also be used to reduce the temperature in the top layer of the stack with a relative low energy cost. But still fans are thought for a 2D system.

As has been shown before, 3D integration needs different techniques specially designed to cool a 3D IC, because the thermal problem has a different nature and has to be addressed differently.

Table 3.2: 3D cooling system summary

	Heat Sinks	Fans	TTSVs	Placement	Thermoelectric	μchannels
Cooling power	✗	?	?	✓	?	✓
Energy consumption	✓	?	✓	✓	✗	?
Technological cost	✓	✓	✗	✓	?	✗
Inner layers cooling effect	✗	✗	?	?	?	✓
Integrability	✗	✗	✓	✓	✓	✓

Thermal Through Silicon Vias, and thermal grids, appear to be a natural solution for distributing heat inside the chip and achieve an homogeneous distribution. However, this kind of solutions are expensive because the integration of TSVs is not an easy process, and requires high technology costs. Besides, the temperature reduction effect that a TTSV has is not very high, it just works as a thermal conductor, that can transport heat from hot areas to colder regions where a heat sink or a different active element could be placed and alleviate the problem.

Placement of functional units is one of the basis of the thermal problem. Functional units dissipate heat in the active layers which gives the possibility of placing them strategically, achieving a good trade off between heat dissipation and performance. The placement, contrary to other solutions that will be later applied on the design flow, has no technological costs and must be done in the design phase. Placement, however, could not be enough to address the thermal problem, and must be taken as the first step of the thermal optimization.

Thermoelectric effect for 3D ICs has not been considered as a good strategy although many researchers are trying to find an application for micro Peltier cells. Their effectiveness lays in their activation time which is very short and, of course, their controllability by tuning the electrical current, which makes them a good dynamic solution. But the difficulties in integrating the cells, their power consumption and the heat that the cell dissipates inside the chip, makes the technique not very popular among the different 3D cooling techniques.

Liquid microchannels have raised the top of the 3D cooling techniques, being the focus of most researches. Microchannels can dissipate fast and efficiently heat from the inside of the 3D stack. But this solution, changes the wafer structure which implies a new design and high technological cost.

In Table 3.2 a summary of the advantages and disadvantages of the different cooling mechanisms for 3D stacks is presented.

3.2. Proposal

In this PhD thesis, many of the above presented techniques will be combined and used at the same time in order to achieve a better thermal behavior of the 3D stack. To this end, the best properties of each system are fully exploited.

However these methods could not be enough to alleviate the thermal problem when the integration scale is high and there are multiple layers in the 3D stack. These thermal drawbacks are more serious in inner layers, where the dissipation ability decreases and where the architectural existing solutions have a great technological cost. Specifically, in order to be able to design feasible many core systems in 3D, the optimization of the placement

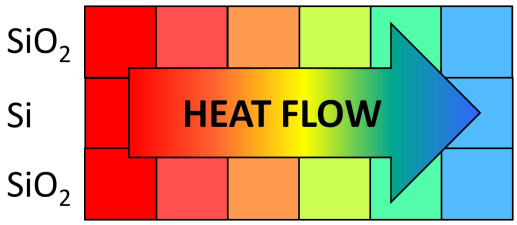
of functional units will be combined with the placement of TSVs and liquid μChannels.

The optimization of the placement of functional units has a major impact on the temperature of the design as has been previously stated, that is why our first step in the optimization process will address this problem. Next, the placement of TSVs in order to guarantee inter layer communication is mandatory. Despite its minimum thermal impact, the optimization of TSVs is of great interest in the minimization of wire length which directly impacts on the performance of the system. Finally, liquid μChannels are deployed all over the stack in order to decrease the possible heat concentration in the inner layers.

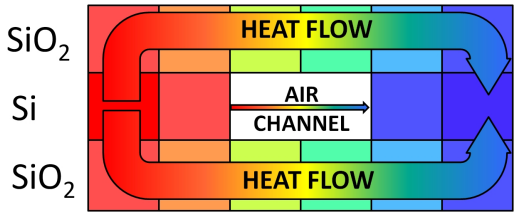
This PhD thesis presents a revolutionary technique which creates isolated thermal regions throughout the stack. Splitting the chip in thermal regions has two main advantages:

1. Since the chip is split, thermal distribution in every domain can be set depending on design constraints. It is easy to set a homogeneous distribution or creating warm and cold regions, which allows to create a better "pre-known" thermal profile in the design phase.

2. If high temperatures are located in certain regions, dissipation mechanisms can be applied in those areas where the thermal problems are exacerbated, minimizing technological costs and optimizing the cooling properties of the different techniques.



(a) Heat path without air channel.



(b) Heat path with air channel.

Figure 3.13: Heat paths with and without air channels.

The method takes advantage of the thermal properties of the air. Since

air is not a very good thermal conductor it can be used as a thermal isolator. Our proposed method consist of creating air trenches in the stack in order to achieve a special isolation in the active layer. Since heat is mainly spread from one cell to its neighbors, inserting an air channel increases the heat path which: 1- reduces the heat transferred from a hot area to a warmer and 2- reduces thermal isolation in the regions defined by the air channels. This effect is depicted in Figure 3.13. If there is not an air channel as in 3.13a, heat is transferred equally through the silicon and silicon dioxide layers. However, when there is an air channel heat is transferred differently because the path is broken. In this case (Figure 3.13b) heat is mainly transferred through the silicon dioxide layers. Air channel also diffuses heat but because of the thermal properties of air this contribution can be almost neglected, however, this contribution is also considered in the thermal simulator which will be later explained in Chapter 4.

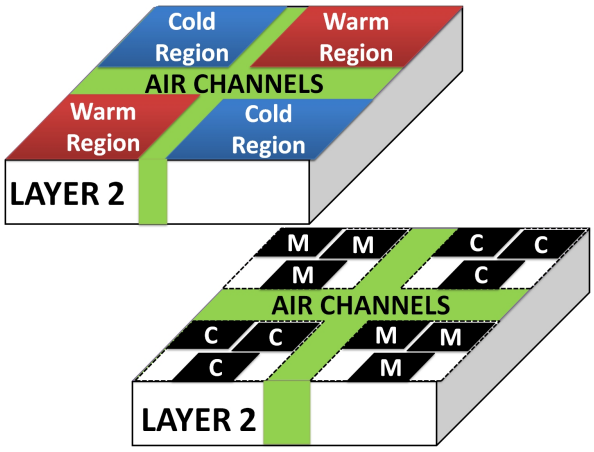


Figure 3.14: Placement of functional units in thermal regions. "M": Memories, "C": Cores.

This revolutionary technique can be boosted if very low pressurized air or even better vacuum is used. When the concentration of air is decreased the thermal conductivity and specific heat also decrease. This means that the technique becomes a more efficient isolation system. However the creation of these channels have to be carefully done taking into account physical stresses. Since the air channels are sealed the creation of vacuum chambers is preferred from the physical and thermal point of view rather than filling them with air. As it is known, gases increase their pressure due to the heat absorbed from the environment following the ideal gas law given in Equation 3.2, which could force inner stresses inside the chip.

$$PV = nRT \tag{3.2}$$

Applying this technique the whole 3D design volume can be horizontally considered as independent regions from the thermal point of view, which makes the activation or optimization of the different cooling techniques easier and more effective, decreasing working energy. I.e, liquid μChannels can be deployed only in the hot regions, reducing the number of routed μChannels, decreasing fabrication and working costs.

As will be then shown in Chapter 5, the results obtained with the combination of our air channels, liquid μchannels, Through Silicon Via and fan coolers outperforms the results obtained by traditional techniques, not only in terms of thermal behavior, but also decreasing the economic cost derived by technological fabrication of cooling systems.

Chapter 4

CAD Solutions for the 3D Thermal Issue

Seamos realistas, pidamos lo imposible
Lema de la revolución del 68

In chapter 3, we have seen many techniques to cope with the thermal problem in 2D and 3D integrated circuits. These techniques offered different solutions that must be considered in the design phase of the integration flow, which means that economic resources to manufacture the stack or to add external elements must be included in the budget.

Although hardware techniques can reduce significantly temperature inside the chip, these techniques must be optimized using different approaches given by optimization algorithms. Since hardware devices imply additional fabrication costs, optimization algorithms can be used in order to reduce these costs without losing effectiveness.

Seeing how the nature works, and the economic repercussions of the not optimized actions, we must develop tools to achieve the best efficiency. These tools are usually based on mathematical formulations of the real problem, where the new wide world of the optimization comes into play.

Simplifying, an optimization problem consists of minimizing or maximizing a mathematical function, choosing among all the possible input values, those which achieve the best output result, in short, finding the *best* solution, from all feasible solutions.

4.1. Thermal Model

Up to this moment, thermal issues and some of the hardware solutions adopted to cope with them have been reviewed, but, in order to better understand the thermal optimization process that will be carried out by CAD

tools, the mathematical method that is used to calculate the temperature distribution throughout the 3D stack must be first introduced. The development of an accuracy mathematical model to calculate the temperature profile of the layers of the 3D chip is essential since the CAD tools use these data as feedback in the optimization flow.

For this purpose a conventional compact model for solids as the one described in [8] will be used. A short description of the method will be explained in the following.

4.1.1. Heat Transfer in Solids. Diffusion

In a multidimensional diffusive heat transfer mechanism we have to define three very important vectors. The first one is equivalent to the scalar temperature ($\vec{T}$), the second is the maximum increase of this vector and is given by its gradient $\nabla \vec{T}$ and the last one is the heat flux denote as $\vec{q}$.

If we take Fourier's law we have that heat flux is directly proportional to the temperature gradient which refers us to Equation 4.1.

$$\vec{q} = k \cdot \nabla \vec{T} \tag{4.1}$$

where k represents the thermal conductivity, which also depends on position and temperature $k = k(\vec{r}, T(\vec{r}\ t))$. Most materials are homogeneous which makes the thermal conductivity dependent on just temperature.

The first law of thermodynamic says that heat transfer rate (Q) comes by the difference in the internal energy (U) by time, which leads to Equation 4.2. If this is applied to a 3D volume as the one depicted in figure 4.1 we can assume that the heat conducted out of the surface (dS) in the region R is $(-k\nabla\ \vec{T}) \cdot (\vec{n}\ dS)$. The heat generated inside the region R has to be added to the last term and we have a total contribution given in Equation 4.3, where the first term on the right side is an integral surface representing the loss of heat from the volume due to heat conduction and the second term is an integral volume representing the rate of generation of heat inside the volume due to conversion from another form of energy (chemical, electrical etc.).

$$Q = \frac{dU}{dt} \tag{4.2}$$

$$Q = -\int_S (-k\nabla\ \vec{T}) \cdot (\vec{n}\ dS) + \int_R \dot{q} dR \tag{4.3}$$

Combining Equation 4.3 with 4.2 and using also Equation 4.1, taking the limit $R \to 0$, applying Stoke's theorem, and assuming that the material has isotropic thermal conductivity, we get that heat diffusion equation in three

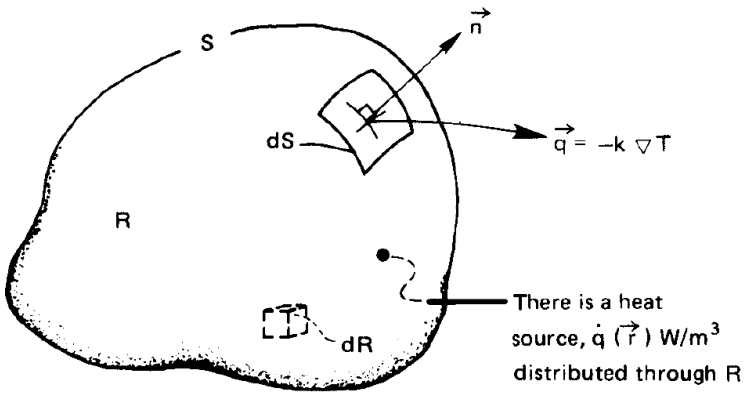


Figure 4.1: Compact 3D solid model [100].

dimensions is given by Equation 4.4, where C_v is the volumetric specific heat of the material and T is the temperature of the volume.

$$-k\nabla^2 \vec{T} + C_v \frac{dT}{dt} = \dot{q} \tag{4.4}$$

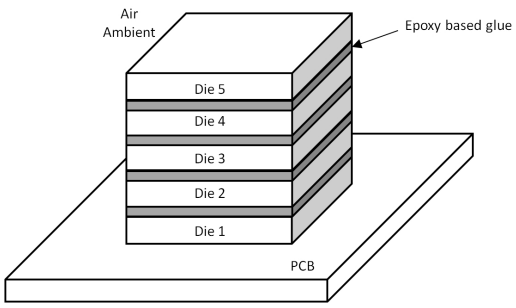
Equation 4.4 can be converted into an ordinary differential equation applying finite difference approximation. The solid volume must be discretized along the three coordinates $\Delta x, \Delta y, \Delta z$. If the temperature in a certain position is given by T_i, j, k, Equation 4.4 can be rewritten as:

$$\begin{aligned} &- k\frac{T_{i+1,j,k} - 2T_{i,j,k} + T_{i-1,j,k}}{\Delta x^2} - k\frac{T_{i,j+1,k} - 2T_{i,j,k} + T_{i,j-1,k}}{\Delta y^2} - \\ &- k\frac{T_{i,j,k+1} - 2T_{i,j,k} + T_{i,j,k-1}}{\Delta z^2} + C_v \Delta x \Delta y \Delta z \frac{dT}{dt} = \dot{q}\Delta x \Delta y \Delta z \end{aligned} \tag{4.5}$$

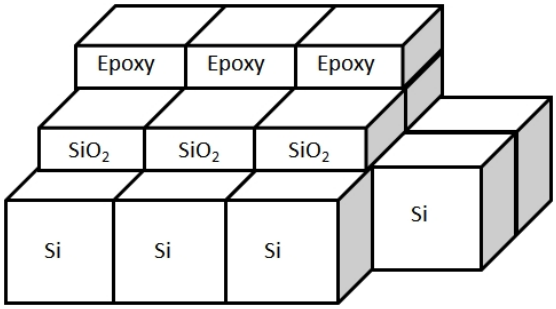
With the above equation we can develop a mathematical thermal/method that will basically consist of splitting the physical stack into small cells and calculating the temperature of every cell that compose the 3D stack.

In order to understand the thermal model the reader is referred to Figure 4.2a, where a schematic view of a 3D stack is shown. In Figure 4.2b we can see how a die is split into small cells.

Within each die, functional units, that dissipate power, and hence heat, are placed in the silicon layers (Si). When a cell is a part of a functional unit it will be considered as a heat source. SiO_2, $Epoxy$ and not dissipating Si cells, are purely diffusive in terms of thermal behavior, that is why the use of Equation 4.5 is valid. Therefore, the heat generated by these heaters flows through the body of the 3D stack until it reaches the edge of the chip



(a) Schematic 3D stack



(b) 3D stack decomposition in small cells

Figure 4.2: Stack decomposition.

where it is spread through natural convection or other dissipating tools such as microchannels or the heat sink.

The dominance of diffusion phenomenon allows us to describe every cell by its equivalence to an electronic Resistance-Capacitance (RC) circuit as described in [68], [139] or [142]. To this end, the whole stack has to be divided into small thermal cells as it is shown in Figure 4.2. Each of these cells, of width (w), height (h) and length (l); constitute the base of the RC model becoming a node of the circuit net as the one published by [8].

The nodes of the net are modeled with six resistances that represent the conduction of heat in all the six directions and a capacitance that represents the heat storage inside the cell. Four resistances, the ones in the same plane, connect each cell to its lateral neighbors (north, south, east and west). The other two resistances connect the cell with the upper and bottom cell respectively (top and bottom), as can be seen in Figure 4.3

The 3D stack is built over a Printed Board Circuit (PCB) surface, that is considered to be adiabatic, this means that no heat diffusion occurs in the bottom layer. The model also considers the heat diffusion to the surrounding environment allowing the simulation of different chip packages by tuning the resistance and conductance parameters of the cells located in the surfaces of the stack.

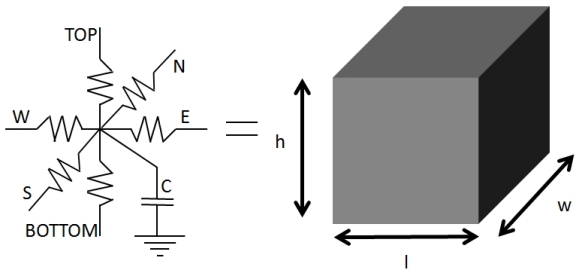


Figure 4.3: RC circuit equivalence for a solid stack cell [37].

Once the whole 3D stack is split into small cells the values of the already built circuit have to be calculated using the following expressions:

$$g_{top/bottom} = k_{th}.\frac{l.w}{(h/2)}, \tag{4.6}$$

$$g_{north/south} = k_{th}.\frac{l.h}{(w/2)}, \tag{4.7}$$

$$g_{east/west} = k_{th}.\frac{w.h}{(l/2)}, \tag{4.8}$$

$$c_{top} = sc_{th}.(l.w.h). \tag{4.9}$$

The subscripts *top*, *bottom*, *east*, *south*, *north* and *west*, indicate the direction of heat conduction. k_{th} and sc_{th} are the thermal conductivity and the specific heat capacity per volume unit of the material, respectively.

All these equations and assumptions are true when the heat transfer process is diffusive. For our experimental results, as will be seen later in Chapter 5, we can use these equations to model silicon, silicon dioxide, epoxy, TSV, and air cells. However in our design there are some cells that are composed by μchannels. These cells transport fluid that is pumped from the outside of the chip, what causes that the heat transfer process is not diffusive anymore. Therefore a different thermodynamic theory must be developed in order to model this process.

4.1.2. Heat Transfer in Fluids. Forced Convection

The explanation of how a fluid moves inside a channel is out of the scope of this study, however, final results and conclusions of the thermodynamical study is presented in the following. For more information the reader is referred to [100].

In the same way it was done with a solid volume, a new region filled with a fluid-flow volume must be defined as the one depicted in Figure 4.4, where a moving fluid with a velocity field $\vec{u}\ (x, y, z)$ is represented.

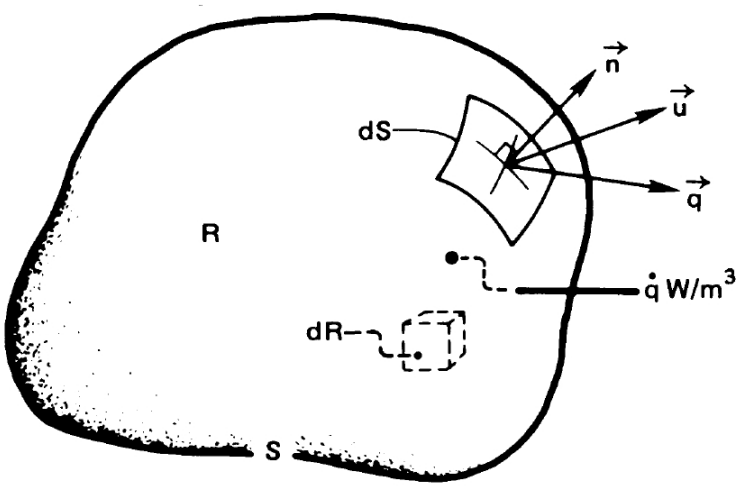


Figure 4.4: Compact 3D fluid model [100].

Some assumptions that are valid when the fluid is moving on our μchannels must be taken:

- The variations in the pressure of the fluid are not large enough to affect the thermodynamic properties of the fluid. This is true because the coolant moving through the μchannel is a liquid which is mostly incompressible.
- It is temperature the only factor that affects the density of the fluid.
- The fluid is supposed to behave as homogeneous.
- The impact of friction in the temperature of the fluid is negligible.

Under these approximations the energy conservation equation for heat transfer in a liquid volume can be then written, similarly to Equation 4.4 as:

$$\int_R \left(\frac{\partial(\rho \hat{u})}{\partial t} + \nabla \cdot (\rho \vec{u}\ \hat{h}) - \nabla \cdot k \nabla T - \dot{q} \right) dR \tag{4.10}$$

where ρ is the density of the fluid, $\hat{u}$ represent the internal energy, and h the enthalpy of the fluid.

The difference between the solid volume and the fluid is that now we have a new term given by the velocity of the fluid $\vec{u}$. As it was done with the solid volume, applying Stoke's theorem in the integral and taking the limit $R \to 0$ we get that:

$$-k\nabla^2 \vec{T} + C_v \frac{dT}{dt} + C_v \vec{u}\ \nabla \vec{T} = \dot{q} \tag{4.11}$$

The Laplacian operator appears again because it has been assumed that heat conduction is isotropic. The new term is introduced because of convection. This term can be calculated, if we consider a cubic cell, as the product of the velocity of the fluid flowing out, the surface temperature, the area of the surface and volumetric heat capacity of the fluid. When the fluid goes in through the surface, the term must be considered negative, on the other hand, when it goes out it is considered as positive.

The equations, as was done before with the solid volume have to be discretized applying finite difference approximation. The expression that characterize the heat transfer process for a liquid in movement cell is presented in Equation 4.12

$$\begin{aligned} &- k\frac{T_{i+1,j,k} - 2T_{i,j,k} + T_{i-1,j,k}}{\Delta x^2} - k\frac{T_{i,j+1,k} - 2T_{i,j,k} + T_{i,j-1,k}}{\Delta y^2} - \\ &- k\frac{T_{i,j,k+1} - 2T_{i,j,k} + T_{i,j,k-1}}{\Delta z^2} + C_v \Delta x \Delta y \Delta z \frac{dT}{dt} + \\ &+ C_v u_{average} \Delta A (T_{S2} - T_{S1}) = \dot{q} \Delta x \Delta y \Delta z \end{aligned} \tag{4.12}$$

$u_{average}$ is the average velocity of the fluid in the cell. Obviously, as the liquid can only flow in one direction through the μchannel, the vector $\vec{u}$ has just one component. It must be noted also, that if the velocity is zero Equation 4.12 equals the equation of the solid cell 4.5, because if the fluid is not in movement heat is then transferred diffusively. A indicates the section of the cell crossed by the liquid, and T_{S1} and T_{S2} are the temperatures of the ends of the liquid cell.

Liquid cells can also be modeled as RC nodes in our electric circuit, however the description of these cells is not as easy as happened with the solid ones. For this end, a schematic view of a liquid channel surrounded by solid cells is depicted in Figure 4.5, where a μchannel flows from south to north following the arrow direction. Therefore, it can be assumed that the dominating heat transfer mechanism in this direction is not diffusion but forced convection, thus, the second term in the discretized equation 4.12 can be neglected in that direction. This means that there will be no resistances in the μchannel direction but this elements have to be replaced by the convection term $C_v u_{average} \Delta A (T_{S2} - T_{S1})$. This convection term depends on the temperature of the surfaces which have an equivalence in our RC model by the use of voltage controlled current sources. The equivalent RC model for a μchannel cell is shown in Figure 4.6

Since the RC model has changed in the μchannel direction we have to calculate the term related to convection. Thus, the convection term of the cell can be calculated as follows [140]:

$$C_v u_{average} \Delta A = C_v u_{average} \Delta l \cdot h = C_v \dot{V} \cdot \frac{1}{A} \Delta l \cdot h \tag{4.13}$$

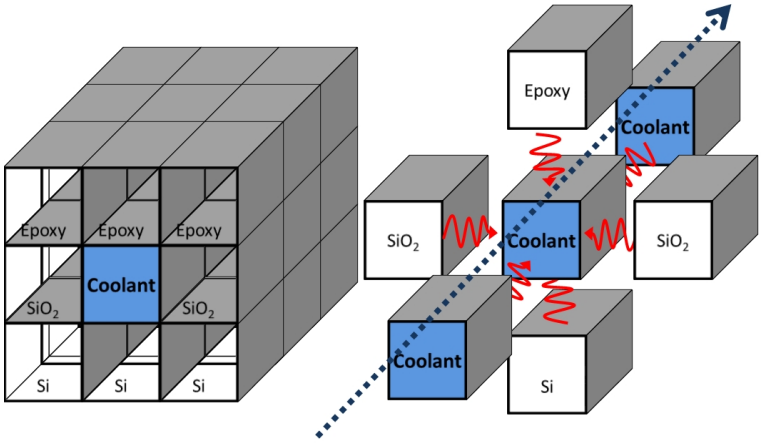


Figure 4.5: Cell decomposition in the case of μchannel.

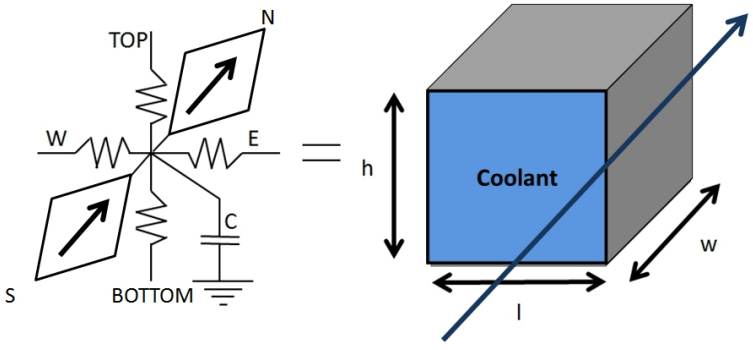


Figure 4.6: RC equivalence of a μchannel cell.

From this equation and applying central differencing method [14] the temperatures of the edges of the cell T_{S1} and T_{S2} can be calculated interpolating the node temperatures of two μchannel cells. If the cell in Figure 4.5 is taken, we can assume that the interface temperature T_{S2} of the central cell can be calculated as $\frac{T^{north}+Tcell}{2}$, and the interface temperature T_{S1} can be calculated as $\frac{T^{south}+T^{cell}}{2}$, where T^x denotes the temperature of the cell. With these parameters already calculated equation 4.13 can be rewritten as:

$$\begin{aligned} C_v u_{average} \Delta A (T_{S2} - T_{S1}) &= C_v u_{average} \Delta A \left(\frac{T^{north} + Tcell}{2} - \frac{T^{south} + T^{cell}}{2} \right) \\ &= \kappa (T^{north} - T^{south}) \end{aligned} \tag{4.14}$$

where $\kappa = C_v u_{average} \frac{\Delta A}{2}$. With the convection term calculated, we have the description of how heat is transported from the south interface to the north edge of the cell, modeling how the temperature of the coolant rises as it goes through the μchannel. To model the whole μchannel the two boundary conditions, temperature at the beginning of the channel (T_{in}), and the temperature at the end (T_{out}), have to be applied.

The first cell of the channel, has its temperature set to a constant given by the pumping system (T_{in}). This fact, sets the first Dirichlet condition at the entrance of the channel. This is very important because the inlet temperature will be used as an input to the recursive model and will be the seed of the convective calculation. The other boundary condition have to be set at the end of the channel. We have to assume that the temperature increment in the last cell of the channel is zero. Thus, $T_{out} - T^{lastcell} = 0$, which set a Neumann boundary condition at the end of the channel.

With these conditions and Equation 4.14, we have modeled the voltage controlled current sources, that means that we have modeled *north* and *south* connections. However, the four remaining conductances that connect the μchannel cell with solid neighbors have to be still calculated. They can be calculated using heat transfer coefficients obtained from either empirical correlations or numerical presimulation, as follows: [140].

$$g_{top,bottom} = h_{f,vertical}(l \cdot w) \tag{4.15}$$

$$g_{east,west} = h_{f,side}(h \cdot w) \tag{4.16}$$

$$c_{top} = sc_{th}.(l.w.h). \tag{4.17}$$

h_f values represent the heat transfer coefficients due to the forced convection that occurs in the μchannel. As can be seen in [100], heat transfer

coefficients in fluids depend not only on the fluid itself but also on other parameters such as velocity or dimensions and shape of the pipe. According to [100] for the proposed model, and considering that heat interchange does not depend on the direction (isotropic fluid), heat coefficients can be calculated with Equation 4.18, where $k_{coolant}$ is the thermal conductivity of the fluid, Nu is the Nusselt number, which is the ratio of convective to conductive heat transfer across the boundary. Parameter d_h, is the hydraulic diameter of the channel that, in our particular case can be described as: $d_h = \frac{2h \cdot l}{h+l}$. As can be seen in Equation 4.17, conductance for the liquid is calculated in the same way as for solids.

$$h_f = \frac{k_{coolant} Nu}{d_h} \tag{4.18}$$

The major difficulty is to determine the Nusselt number. Typically, for free convection, the average Nusselt number is expressed as a function of the Rayleigh number and the Prandtl number, and it is written as: $Nu = f(Ra, Pr)$. However this is not our case, because the coolant is pumped. For forced convection, the Nusselt number is generally a function of the Reynolds number and the Prandtl number $Nu = f(Re, Pr)$. This number is generally determined empirically [79]. Under our μchannels conditions, where there is a low aspect radio of the channel, and where radial isothermal conditions are assumed, the Nusselt number can be expressed as follows: ([133],[135])

$$\begin{aligned} Nu = 8.235(1 - 2.0421AR + 3.0853AR^2 - 2.4765AR^3 + \\ + 1.0578AR^4 - 0.1861^5) \end{aligned} \tag{4.19}$$

where AR refers to the aspect radio of the μchannel, in our particular case $AR = \frac{h}{l}$. From this expression it can be derived that, as long as the coolant flows below the limit of turbulent regime, the Nusselt number, and consequently the heat transfer coefficient h_f, does not depend on the velocity of the coolant.

4.1.3. Heat Transfer in Air Channels. Diffusion, Natural Convection and Radiation

In the particular case of air channels we have to taken into account several heat transfer mechanisms. Opposite to what happened in solid and fluid convection and radiation are no longer negligible. When heat is transferred through an air channel all the three phenomena must be considered.

From the situation for unidimensional heat transfer depicted in Figure 4.7, Equation 4.1 becomes

$$\frac{d^2T}{dx^2} = 0 \tag{4.20}$$

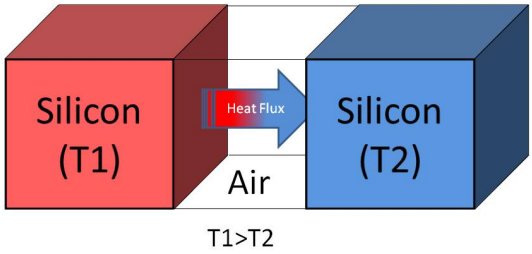


Figure 4.7: Heat transfer through an air channel.

which general solution is $T = Ax + B$. Bound conditions have to be fixed to solve A and B. In $x = 0$, the temperature is T1, and when $x = L$ temperature is T2. With these conditions temperature is given as:

$$T = \frac{T2 - T1}{L}\Delta x + T1 \tag{4.21}$$

Equation 4.21 reflects the diffusion mechanism but still, convection and radiation have to be calculated. For the convection case, orthogonal to liquid channels characterization, the convection mechanism is given by natural convection which means that new equations have to be formulated.

Convection transfers heat faster than conduction. As happened with the forced convection that ocurred in our μchannels, the main problem of convection calculus is that the Nusselt number has to be calculated experimentally. Nusselt number will be calculated using Rayleigh and Prandtl numbers for the air, and relating them using Churchill and Chu equation:

$$Nu = \left(0.85 + \frac{0.387\Delta Ra_L^{1/6}}{[1 + (0492/Pr)^{9/16}]^{8/27}}\right)^2 \tag{4.22}$$

Prandtl number for air is 0.707 ([43]), and Rayleigh number is described by:

$$Ra = \frac{g\Delta\beta}{\nu\Delta\alpha}\Delta\Delta\Delta T \tag{4.23}$$

where β is the thermal expansion coefficient, ν is the kinematic viscosity, L is the characteristic length and α is thermal diffusivity. For air at 20°C Grashof number is $Gr = 78 \cdot 10^9 \Delta\Delta\Delta T \Delta L^3$.

Using these parameters in our study we can calculate heat transferred by convection with Equation 4.24.

$$\dot{q}_{conv} = Nu \cdot \dot{q}_{cond} \tag{4.24}$$

Additional to convection and diffusion there is another contribution to heat flux is given by electromagnetic waves. It is produced at atom level, when electrons change their energetic levels. It is well known that these electromagnetic waves depend on the temperature of the object.

Radiation opposite to conduction or convection, does not need a material to spread heat, so it can happen in the vacuum. As it is an electromagnetic wave it travels at the speed of light so it is the fastest way to spread heat, however it is difficult to design a good radiator to transmit heat.

In our particular case we are going to suppose that our silicon walls behave as a black body, so its maximum power of radiation will be described by the Stefan-Boltzmann law:

$$\dot{q} = \sigma \Delta T_s^4 \tag{4.25}$$

where σ is the Stefan-Boltzmann constant; $\sigma = 5.67 \cdot 10^8 W/(m^2 \Delta K^4)$ and T_s is the temperature of the emission surface.

4.1.4. Thermal model solution

In the above paragraphs the mechanism that describes heat transfer in a 3D stack has been presented. It has been also shown how the physical layers of the stack are decomposed into small cubic cells, showing the analogy between physical thermodynamic principles and electric circuit theory. With Equations 4.6-4.9 and 4.15-4.17, the values of the resistances and capacitance for each cell can be calculated and a set of equations, one for every unitary cell, is created.

The behavior of the resulting RC circuit can be described using a set of first order differential equations via nodal analysis [151] as follows:

$$GX(t) + C\dot{X}(t) = BU(t), \tag{4.26}$$

where $X(t)$ is the vector of cell temperatures of the circuit at time t, G and C are the conductance and capacitance matrices of the circuit, $U(t)$ is the vector of input heat (current) sources and B is a selection matrix. G presents a sparse block-tridiagonal matrix and C would exhibit a diagonal structure if there were only solid cells, but the convective nature of heat transferred between two neighbor cells of the μchannel adds some non diagonal elements to the matrix. The direction of the flowing coolant makes C matrix no symmetric because heat is transferred along the direction of the channel which

breaks the existing symmetry when only solid cells were considered in the stack.

In addition, G and $U(t)$ are functions of the cell temperatures $X(t)$, making the behavior of the circuit non-linear. This is because of the temperature-dependent thermal conductivity of silicon. It has been assumed a first-order dependence of these parameters on temperatures around 300K.

The most important parameters used in the simulation are shown in Table 4.1.

Si linear thermal conductivity	295 W/(mK)
Si quadratic thermal conductivity	-0.491 W/(mK^2)
SiO_2 thermal conductivity	1.38 W/(μmK)
Si specific heat	1.628 x 10^6 J/m^3K
SiO_2 specific heat	4.180 x 10^6 J/m^3K
Isolating Air thermal conductivity	2.4 x 10^-3 W/(mK)
Isolating Air specific heat	1 x 10^4 J/m^3K

Table 4.1: Thermal properties of materials.

Once the values of the resistances and capacitances for each cell of the 3D stack have been calculated a set of equations describing the steady state of the system is built. Therefore, the equation set to be solved is described by:

$$GX = BU. \tag{4.27}$$

Since our study is focused on the steady state, the transient term given by $C\dot{X}(t)$ in Equation 4.27 disappears. However, if the study of the transient situation is required, a different mechanism to solve the equation set have to be implemented to integrate numerically the system with mathematical methods such as Forward Euler or Backward Euler.

The above set of steady state equations is solved by the inversion of the matrix G using the sparse LU decomposition method [42]. Since the system to be solved is non linear, and we have to take into account the input sources of the functional units, the equations have to be solved repeatedly. In the solving loop the matrices are updated after each iteration. Initial conditions set every cell to the ambient temperature as the seed of the model. In each iteration temperatures for the whole 3D stack are calculated until convergence is reached. The description of this iterative algorithm is described in Algorithm 4.1 [8].

This thermal model has been tested and validated in [140] where the model is embedded in a complete 3D thermal simulator, 3D-ICE ([49]). This 3D thermal model has been modified to include the air channels topology combined with TSVs and μchannels.

Algorithm 4.1 Calculation of the steady state temperatures of the 3D stack.

1. Define:
   X^r= vector of cell temperatures during the r^{th} iteration,
   G^r= conductance matrix during the r^{th} iteration,
   U^r= input vector during the r^{th} iteration.
2. Set r=0. Generate an initial-guess for X^0
3. Calculate G^0 and U^0 using the initial-guess X^0
4. $X^{r+1} = (G^r)^{-1}BU^r$
5. Calculate G^{r+1} and U^{r+1} using the updated temperatures X^{r+1}
6. If $\left\|X^{r+1} - X^r\right\|$<(a predetermined error criterion), exit. Else set r=r+1 and go to step 4.

4.2. 3D Thermal Optimization Algorithms

In this section a review of the state of the art of the optimization algorithms used to improve the thermal profile in integrated circuits will be presented. This problem has been widely studied since the moment that some studied proved that some reliability problems were related to temperature, and these problems were becoming more and more common.

The most part of the algorithms that coped with the problem of the temperature in 3D are a natural evolution of those used to optimize the thermal problem in classical 2D ICs. These algorithms were focused on optimizing the placement of the functional units at micro architectural level.

One of the first works in thermal aware placement is the one by [85]. In this work, a placement scheme that considers both electrical performance requirements and thermal behavior for the high-performance multichip modules is described. Practical thermal models are used for placement of high-speed chips in multichip module packages under two different cooling environments: conduction cooling and convection cooling. Placement methods are modified to optimize conventional electrical performance and chip junction temperatures. As has been already proven, placement solutions are no longer suitable for 3D integration systems, because these techniques are not able to alleviate the high temperatures caused by the integration and the difficulty of inner layers to dissipate power to the environment.

Temperature-tracking is thus becoming of paramount importance in modern Electronic Device Automation (EDA) tools. These methods are aimed at reducing hot spots in a design without compromising traditional design metrics such as area and wire length. Some works such as [23] or [147], convert the user-specified temperature constraint into the corresponding power distribution constraint as an alternative placement objective.

A very important revolution in the field of temperature optimization was

the development of a well known thermal simulator, "HotSpot"by the Department of Computer Science of the University of Virginia. This thermal simulator is perfectly described in [84] and [72]. From this moment on, many studies used HotSpot to run thermal simulations and the articles published ,related to thermal optimization, raised considerably. This simulation platform has evolved in order to include 3D designs but still presents some deficiencies like the inclusion of TSVs or μchannels.

The different state of the art found in the literature, admits several divisions according to their functionality, representation or optimizing algorithm which will be considered as the main division in this PhD Thesis.

Among all the techniques used to solve the thermal aware placement problem in 2D ICs the ones that have become more popular have been linear programming, Simulated Annealing (SA), and evolutionary algorithms. Obviously these algorithms have evolved to the 3D problem.

Apart from these design techniques other software approaches have been taken in order to reduce the temperature inside the chip. These different techniques are based in operative system routines which control the workload of the the cores by migrating tasks from one core to another, and other software techniques that control the voltage and the frequency of the system.

Although explaining all these techniques in detail is out of the scope of this PhD Thesis, in the following, we will briefly review the most important works that carried out thermal optimizations using these mathematical tools. For some more information about these mathematical techniques the reader is referred to Appendix A.

4.2.1. Linear Programming (LP)

Linear programming is a mathematical method to calculate the best result (maximum or minimum depending on the problem) in a mathematical model that must formulated using linear relationships. Mathematical optimization is a wide field of investigation and Linear Programming is just a specific case.

Following this methodology we can find several works focused on the thermal aware design. One of the first studies that use Linear Programming as the base of their mathematical optimization is the work carried out by Healy et al. [67]. This paper presents the first multiobjective microarchitectural floorplanning algorithm for high-performance processors implemented in 2D and 3D ICs, as a natural evolution of the algorithm, but, as the first thermal aware optimizations did, it only took into account the placement of functional units.

The optimization process considers not only the power dissipated by the functional units in the chip, but also other design objectives such as the area and wirelength. The 3D floorplanning algorithm takes into account that heat

is spread horizontally but it also computes how heat is transferred from one layer to others. Despite the fact the paper exhibits good results achieving a trade off between performance metrics as wirelength and temperature, it does not consider the placement of TSVs critical for the 3D design, or how temperature is distributed in the layers which implies the appearance of thermal gradients in one layer and between layers. These thermal gradients, as was seen in Chapter 2, have a great impact in reliability.

In thermal aware floorplanning using LP programming the problem basically consist only of optimizing the positions of the functional units to achieve a better thermal distribution, decreasing thermal stress which affects the reliability of the chip.

Authors as [97] present a incremental floorplanner using Mixed Integer Linear Problem (MILP). The algorithm adopts incremental changes to eliminate hotspots, using three different placement algorithms: migrating computation, growing unit and moving hotspots. Three different algorithms based on the placement of functional units. It achieves great results in terms of temperature distribution and computation time, however important design constrains in 3D technology as TSVs are again obviated.

The incorporation of TSVs in the optimization loop is made in [98]. In this work, incorporating thermal vias into 3D IC is used as a promising way to mitigate thermal issues by lowering down the thermal resistances between device layers, which means that TSVs are used as Thermal Through Silicon Vias. The study is similar to other works since the optimization is based on allocating functional units in certain positions in order to reduce the thermal impact in the chip. However Li et al., re-allocate white space in the 3D IC to facilitate via insertion. Although this study starts to consider two different ways of facing the thermal problem, it does not consider both variables inside the optimization process which could be a problem if the remaining white spaces in the chip are not enough to route vias.

MILP has proven to be an efficient and fast approach. However, when MILP is used for thermal aware floorplanning, the (linear) thermal model must be added to the topological relations and the resultant algorithm becomes too complex [48], specially as the problem size (number of cores, in our case) increases. In this PhD thesis we have developed several floorplanning models using MILP. To this end, we have developed promising linearization of objetive functions and novel algorithms to place TSVs [36]. However, we concluded that MILP is not scalable and moved to other directions like evolutionary computation.

4.2.2. Simulated Annealing

Simulated annealing (SA) is a probabilistic metaheuristic method for finding optimum solutions which search in the entire space of search and

moves towards the optimal solution.

Simulated Annealing has been widely used in the 2D floorplanning problem. In 1989 Rutebar, in [128] made an overview of the simulated annealing algorithms where the goal was mainly to reduce the area of the chip, and with this shrinking process the first thermal problems came into play.

From this moment on, new alternatives had to be risen. First thermal aware algorithms for 3D ICs, were developed as a natural evolution of those used for 2D. For example, Cong et al. [29], published in 2004 a thermal aware floorplanner using simulated annealing that was based on those published for the 2D case. This work is considered to be one of the pioneers in the thermal aware floorplanning field. In this paper two major contributions that exposed and inspired the use of different algorithms to exploit the thermal aware floorplaning. The first contribution is the new representation used for the solutions which are based on combined arrays, and the second one is the inclusion of the resistive network for the thermal model in the algorithm which is translated into a reduction in computation time.

Following the investigation line established by Cong other authors such as Healy [67] or Hung [75], use different simulated annealing based algorithms to optimize thermal distribution in 3D stacks. In the former Healy et al. as was presented before, used a combined algorithm using not only simulated annealing but also LP. The latest is more focused on alleviating the power dissipated by connections minimizing the wirelength and hence, has to optimize the placement of the functional units in the stack. These two works, as happened with the first studies using LP, do not consider TSVs in the optimization flow which are to be a very important element in 3D technology.

In [155], the consideration of TSVs becomes quite clear. Wong et al. use a tree representation for the solution in order to improve performance over traditional 2D circuits by reducing wirelength and interconnect delay. This paper presents a thermal via insertion algorithm that can be used to plan thermal via locations during floorplanning. The thermal via insertion algorithm relies on a thermal analyzer based on random walk techniques. In this paper the insertion of TTSVs is considered from the beginning and it is a part of the whole optimization problem making the formulation quite complete. However, as all the previous reviewed works does not consider additional cooling methods, restricting the optimization problem to the placement of functional units and TSVs.

One of the major problems that simulated annealing exhibits is the representation of the solution. As can be see in A.2 the algorithm is very sensitive to the set up of the algorithm itself and the representation of the solution. This fact has opened a whole new line of investigation for researchers based on finding better representations.

4.2.3. Evolutionary Algorithms (EA)

Evolutionary computation, has its origin in the 50s and 60s when some independent researchers started to study evolutionary systems in the nature, based on the theory developed in 1859 by Charles Darwin [40]. They realized that these natural phenomena could be applied mathematically, optimizing engineering problems following the principle "as nature evolves, specimens have to adapt themselves to new environments". These natural methods were then modeled, and mathematical operators, based on natural selection and mutation where raised as the base of the evolutionary algorithms.

In the literature many floorplanning examples using evolutionary algorithms can be found for the 2D. As in other algorithms the natural evolution of the 2D cases has been used a the base of the 3D floorplanning with the inconveniences that have been exposed before. For example, works as [159], where authors use a hybrid algorithm of genetic and simulated annealing, in order to find an optimum floorplan distribution for dedicated hardware in a short time, is difficult to extend to the 3D case since it does not consider 3D variables as the area and the possibility of more functional units in the chip.

In other studies the extension to the 3D technology is made using simple genetic algorithms as in [76]. Hung et al. present a genetic algorithm for 3D technology. The proposed algorithm tries to distribute temperature evenly across the chip, and optimizes chip area. In order to run the thermal optimization the authors consider a slicing tree representation for the chromosome represented by Polish expression, that by its nature, compacts the area in which functional units are disposed which is translated into an increase in power density and hence, temperature across the 3D stack. Besides the thermal model is not considered in the optimization loop and only favoring those floorplans with a penalty equation in the fitness function. Afterwards in [75], these same authors consider further implications in 3D designs such as the heat due to interconnections, but using simulated annealing for this purpose, but not considering the existence of TSVs in the design.

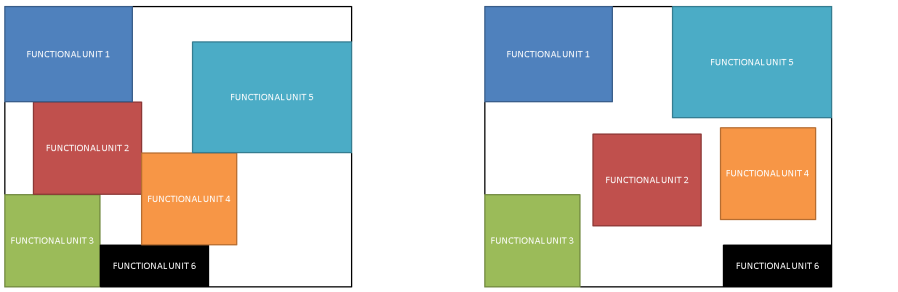
From the computational point of view, VLSI floorplanning is an NP-hard problem. In the work carried out by Tang and Yao [146], a memetic algorithm (MA) for a non-slicing and hard-module VLSI floorplanning problem is presented. This new algorithm uses a genetic search method in order to explore the whole search space in a very efficient way and using an O-tree representation which has been proven to be a very good implementation for area and wire length optimization. Although the algorithm shows very good results in terms of area and connection length it does not take into account the impact of the temperature on the reliability of the chip.

Some other studies such as [28], take a thermal-aware perspective integrating in the optimization flow Through Silicon Vias. The optimization is based on the algorithm developed for 2D ICs, modified in order to consider the natural 3D evolution. The algorithm in particular, transforms a 2D

placement with good wirelength to a 3D placement, with the objectives of half-perimeter wirelength, TSV number and temperature. Although the existence of the TSVs is considered in the study the allocation of these items is not considered in the proper placement optimization and the final floorplan is reached through a series of refinements.

All the above presented studies cover a very important part of the thermal aware design phase, however many other techniques such as liquid μchannels are not covered in depth in the literature when they are combined with placement of functional units or TSVs.

Another existing drawback in the field of evolutionary and simulated annealing methods is the representation of the solution. The existing representations for the floorplanning problems encode the solutions by neighbors in order to save space, however our proposal as depicted in Figure 4.8, is able to allocate the functional units in empty space without the need of placing them next to other functional units. This is vital when the floorplanning has to be done from the thermal aware point of view.



(a) Traditional Neighbors Representation

(b) New Proposals Without Neighbors Representation

Figure 4.8: Different Representation Results.

4.2.4. Execution Time Techniques

Apart from the optimization algorithms presented above, other ways of coping with the thermal problem in multi core systems, and specially in 3D integration technology, have raised as promising techniques to complement thermal-aware floorplanning during the design phase. These techniques consist of routines that are controlled by the operative system which means that they are not used during the design process but in execution time. Many studies in this field can be found in the literature, in the following we will present some studies based on load balancing and Dynamic Voltage and Frequency Scaling (DVFS) methods.

Load balancing consists of adjusting the work load of the processors to

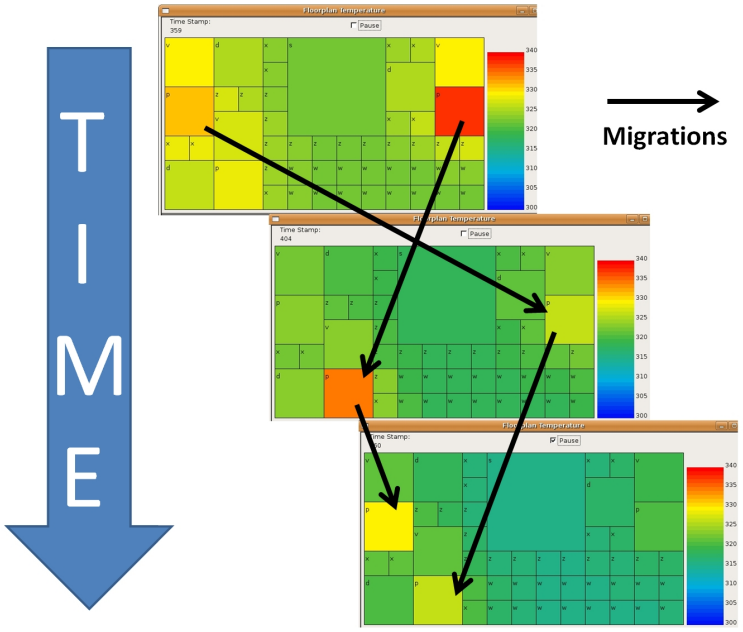


Figure 4.9: Temperature distribution in a 3-core system.

achieve an homogeneous thermal distribution throughout the chip. It is well known that the power consumption of a core is directly proportional to its load. Migrating tasks among cores equilibrates the work load and hence the temperature. A representative image of how the temperature varies due to the migration of some tasks between the processors, represented by letter p, can be seen in Figure 4.9, from [34].

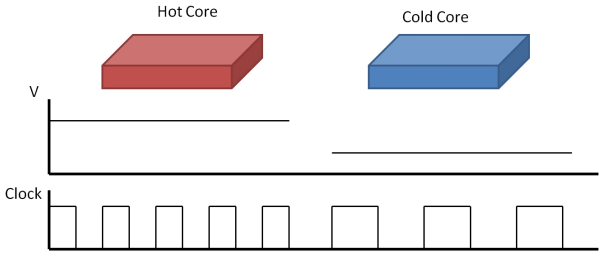


Figure 4.10: Voltage and clock cycle are proportional to the power dissipated by a functional unit.

On the other hand Dynamic Voltage and Frequency Scaling consists of adapting the frequency and/or the voltage of a functional unit or a set of them in the chip. Power consumption, depends not only on the work load, but also in these two electrical parameters, voltage and the clock cycle. By the tuning of these values, a desired thermal pattern can be obtained, but these techniques need to be applied very carefully because they strongly

impact on the performance of the system.

Load balancing techniques have been studied for general purpose parallel computers in the last decade [143, 19]. However, embedded systems and MPSoCs impose constraints, like the low-cost packaging and the portability.

Many approaches have been followed in order to cope with the thermal problem using workload information. Some authors as Nollet et al. [116], proposed a technique that uses the debug registers of the processor to get the system workload information. Therefore, the initial overhead of a heterogeneous MPSoC task migration is diminished by considering these hardware devices which are not always available in current architectures.

Other software techniques make the programmer the responsible of the migration process ([12]). The architecture used in the work of Bertozzi, was composed of one master and an arbitrary number of slaves cores. Even though this paper shows interesting results for such specific architecture, the technique has some drawbacks, because the application have to be programmed for a special hardware.

Götz et al. [63] present a design flow for dynamic relocation of hybrid tasks. These tasks may be executed either in hardware or software and are represented through a *state transition graph*, where each state is known as a computation block and stands for a given task operation.

Most authors, do not take into account that task migration implies some overhead due to the swapping operations. This is not the case of the study carried out by Barcelos which extends the one by Brião et al. [16] who takes into account the task migration overhead in a dynamic environment and discusses its impacts in terms of energy, performance and real-time constraints for MPSoCs based on Network on Chips (NoCs).

However, all these works are not focused in the optimization of the temperature in the systems. More recent approaches, focused in the optimization of the temperature of the chip have been proposed. Donald et al. [46] introduced several thermal management policies such as dynamic voltage and frequency scaling (DVFS) and thread migration based on current temperature, but their work does not consider the thermal history of the cores. This information gives a meaningful information about the future behavior of the system and can be exploited to improve the results of the migration.

Many other works, as the one by Puschini et al. [123], also manages dynamically the voltage and frequency assignment of each core based on game theory. However, the DVFS as a thermal optimization technique is limited by its implementation and its impact on performance.

On the other hand, Powell et al. [62] described techniques that, using the information provided by performance hardware counters, tried to balance the temperature by thread migration. However, it is considered that performance counters do not represent accurately the thermal profile.

In [157], Yang et al. showed an execution ordering approach that swaps

hot and cool threads in cores to control the temperature. This can only be applied once the application has been profiled. Finally, a recent work by Yeo et al. [158] presented a temperature-aware scheduler based on thermal grouping of the applications using a K-means clustering. This work provided interesting results but requires a very complex analysis phase, which grows largely in complexity with the number of considered cores.

In [112] a heuristic optimization for thermal balancing in MPSoCs that adapts the current workload of the cores using DVFS and task migration is proposed. The method is based on the inspection of the standard deviation of the hottest and coldest cores at each moment in time during the execution. Although it shows clear benefits for thermal balancing with respect to previous thermal runaway approaches [46], it can still produce significant thermal unbalance in non-stable working conditions. (i.e., periods of small tasks being executed in the MPSoC or tasks being stop due to I/O processes) because it does not take into account the recent thermal history of the system but just the instant thermal unbalance.

Task migration and DVFS have demonstrated that can improve significantly the thermal profile of a chip. In [34], some migration policies are proposed. These policies optimize the thermal profile of MPSoCs by balancing dynamically the weight of the on-chip thermal gradients, maximum temperature and effect of underlying floorplan on heat dissipation properties of each core. Moreover, the proposed policies are able to minimize the risk of system failure by the minimization of temperature-driven reliability factors, as it considers thermal unbalance in time and space, as they keep a history of the thermal profile of the target MPSoC, which minimizes the number of task migrations.

4.3. Proposed Optimization Algorithm

As it has been exposed, the state of the art present some deficiencies that our proposal fills. This PhD Thesis makes a contribution to the thermal aware techniques combining the optimization of several methods such as the placement of functional units, the placement and number of liquid μchannels, the wire length by the placement of TSVs and a new technique that integrates air channels inside the stack in order to create temperature regions.

For this purpose a software floorplanner has been developed. The optimization phase of our floorplanner is carried out by a Multi-Objective Evolutionary Algorithm (MOEA), that will be called Multiobjective Floorplanning Algorithm (MFA). In Appendix A.3 a description of how a Evolutionary Algorithm works is shown, however, MOEAs have some peculiarities that have to be explained. In this section it will also be shown why, MOEA is the best approach to develop our optimized floorplanner.

The use of a MOEA is proposed because these algorithms have demons-

trated to be an efficient method to solve NP-hard problems. MOEAs are stochastic optimization heuristics where the exploration of the solution space of a certain problem is carried out by imitating the population genetics stated in Darwin's theory of evolution, as every GA does. Selection, crossover and mutation operators, derived directly from natural evolution mechanisms, are applied to a population of solutions, thus favoring the birth and survival of the best solutions. MOEAs have been successfully applied to many NP-hard combinatorial optimization problems. MOEAs encode potential solutions (individuals) to a problem with strings (chromosomes), and combine their codes and, hence, their properties. In order to apply MOEAs to a problem, a genetic representation of each individual has first to be found. Furthermore, an initial population has to be created, as well as defining a cost function to measure the fitness of each solution.

As a second step, the genetic operators have to be defined. These operators will allow us to produce a new population of thermal-aware floorplaning solutions from a previous one, by capturing the interdependencies of the different topological constraints working concurrently. Then, by iteratively applying the genetic operators to the current population, the fitness of the best individuals in the population converges to targeted solutions, according to the metric/s to be optimized and the weight of each of them. For an overview of MOEAs the reader is referred to [27].

For this particular floorplanner a Non Dominated Sorting Genetic Algorithm (NSGA) is proposed. The NSGA is a Multiple Objective Optimization algorithm and is an instance of an Evolutionary Algorithm from the field of Evolutionary Computation. NSGA is an extension of the Genetic Algorithm for multiple objective function optimization. It is related to other Evolutionary Multiple Objective Optimization Algorithms such as the Vector-Evaluated Genetic Algorithm (VEGA), Strength Pareto Evolutionary Algorithm (SPEA) and Pareto Archived Evolution Strategy (PAES). There are two versions of the algorithm, the classical NSGA and the updated and currently canonical form NSGA-II.

The objective of the NSGA algorithm is to improve the adaptive fit of a population of candidate solutions to a Pareto front constrained by a set of objective functions. NSGA uses the same set of operators that any evolutionary algorithm does, that is selection, genetic crossover and genetic mutation.

The population used by the algorithm is sorted into a hierarchy of sub populations. These subsets are created based on the ordering of Pareto dominance. Similarity between members of each sub-group is evaluated on the Pareto front, and the resulting groups and similarity measures are used to promote a diverse front of non-dominated solutions.

For the purpose of this PhD Thesis, the optimizer will follow the basis of a NSGA-II. Algorithm 4.2 shows a pseudo code for an NSGA-II. The *SortByRankAndDistance* function orders the population into a hierarchy of

non-dominated Pareto fronts. The *CrowdingDistanceAssignment* calculates the average distance between members in each front. The *GeneticOperators* function performs the classical crossover and mutation genetic operators of the MOEA. Both the *SelectParentsByRankAndDistance* and *SortByRankAndDistance* functions discriminate members of the population first by rank (order of dominated precedence of the front to which the solution belongs) and then distance within the front (calculated by *CrowdingDistanceAssignment*).

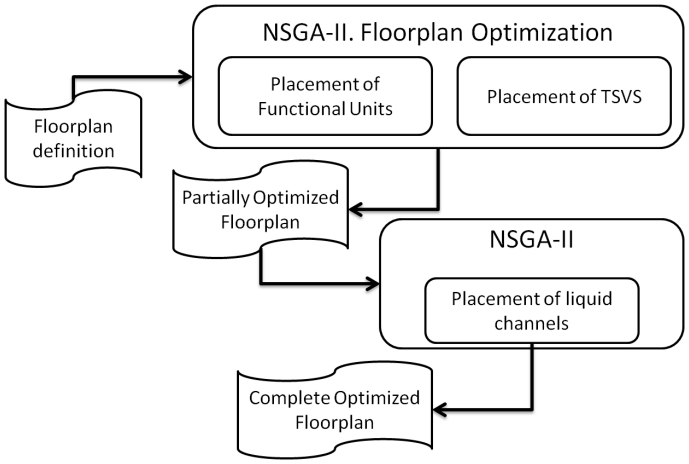


Figure 4.11: Optimization flow.

Once, the basis of the optimization algorithm has been presented, we will show the optimization flow for our thermal aware incremental floorplanner. The proposed optimization flow is depicted in Figure 4.11. First of all a seed design has to be defined to be the input of the optimization process. This initial design is composed by the description of the functional units in which both physical characteristics (width and length) and power consumption data are specified. As additional information, a file, describing the interconnection network of the functional units must be included with the initial design.

Taking the description file as an input to the optimization flow, the algorithm will carry out the optimization of the placement of functional units and TSVs. The algorithm will analyze the whole design volume in order to allocate the functional units in the optimum positions from the point of view of thermal behavior. TSVs are also inserted in this phase. The algorithm makes sure that there is enough room for allocating TSVs fulfilling communication requirements. After this phase the algorithm returns a set of partially optimized floorplans that will be passed as inputs for the next phases.

Finally, μchannels have to be added to the 3D design. This final step, is done by an independent evolutionary algorithm, which has the chosen optimized designs with the TSVs as inputs, and as output, the final optimized

Algorithm 4.2 Pseudo code for a NSGA-II algorithm. [17]

```
1: Input: Population_size, ProblemSize, Prob_crossover, Prob_mutation
2: Output: Solutions
3: Population ← InitializePopulation(Population_size, ProblemSize)
4: EvaluatedPopulation ← Evaluate(Population)
5: NonDominatedSort(EvaluatedPopulation)
6: Selected ← SelectParentsByRank(EvaluatedPopulation, Population_size)
7: Children ← GeneticOperators(Selected, Prob_crossover, Prob_mutation)
8: while StopContion != 0 do
9:    Evaluate(Children)
10:   Union ← Merge(EvaluatedPopulation, Children)
11:   Fronts ← NonDominatedSort(Union)
12:   Parents ← 0
13:   Front_L ← 0
14:   for Front_i ∈ Fronts do
15:      CrowdingDistanceAssignment(Front_i)
16:      if Size(Parents)+Size(Front_i) >Population_size then
17:         Front_L ← i
18:         Break()
19:      else
20:         Parents ← Merge(Parents, Front_i)
21:      end if
22:   end for
23:   if SizeParents) <Population_size then
24:      Front_L ←SortByRankAndDistance(Front_L)
25:      for Parent_1 to Parent_Population_size do
26:         Parents←Parent_i
27:      end for
28:   end if
29:   Selected ← SelectParentsByRankAndDistance(Parents,Population_size)
30:   Population ← Children
31:   Children ← GeneticOperators(Selected, Prob_crossover, Prob_mutation)
32: end while
33: return Children
```

stack including all the mechanisms proposed in this thesis to cope with the thermal problem.

The optimization phases are commented in more detail in the following, starting with the codification and the representation of the chromosomes for the optimization of the placement of the functional units and included in the developed algorithm. Next, the optimization of the TSVs is presented, explaining the optimization process for the placement of TSVs and finally the algorithm which optimizes the distribution of μchannels close the chapter.

4.3.1. Placement of Functional Units

The placement of the functional units is the first step towards the final optimized design. To this end, a technological design must be defined, setting the area, the number of the layers, the previously designed regions separated by air channels and the set of functional units. From this initial design, a representation of the solutions (individuals) has to be studied.

The representation of the individual is extremely important. If the encoding of the solution is good, the algorithm will be able to find optimized solution in less time. The representation is linked to the nature of the problem. If the solution is based on a *yes* or a *no*, it may be obvious that the chromosome has to be binary encoded. On the other hand, if the problem consists of a discretized problem the solution can be encoded with integer numbers.

4.3.1.1. Encoding

Many studies have been done about the encoding of solutions. For example, float point representation has the strong point of high precision and facilitating search in high-dimension space. It is superior to other representation in function optimization and the definition of constraints is easier. However, the search space becomes prohibitive, making the algorithm slow and inefficient. A review of the formulation of problems using real coded genetic algorithms is done in [121]. Opposite to this approach, some authors have based their optimization studies using integer representation of solutions. In this case the space of search is much more restricted and the algorithm can find solutions much faster if the problem does not require such a high precision or if its solution nature is purely integer. In this way, we can find in [61], how the power production of a electric central is optimized. The use of integer codification reduces the size of chromosomes and computation time significantly, and in most cases in which integers are a good choice, an integer codification finds the same optimal solution than a floating point codification. This also happens in [2], where integer encoding is used to obtain accurate and parsimonious sinusoidal representation of signals, or [154] where the integer solution reflects the optimization in the number of antennas

and the rows of them to obtain the optimum Pareto front that achieves low beamwidth and side lobe level.

Nevertheless, the most used encoding has always been binary because of its simplicity to be processed and because the evolutionary operators can be described in a very easy way. Some examples can be found in [152] where a binary-encoded genetic algorithm is used to improve wind turbine models to take advantage of the wind energy, or [57], where the binary encoded algorithm searches the optimum parameters used in the training process of support vector machines for regression.

Other recent studies have gone one step forward and have mixed two or more encoding techniques as [56] which uses a switching representation to solve the knapsack problem, or have modified the traditional way of encoding the solutions as happens in [99] where binary code is modified by splicing or decomposing the chromosome to describe potential solutions of the problem with different precisions. Jelodar et al. in [83], presented a novel binary representation whose length is intelligently adaptable to the given problem.

4.3.1.2. Floorplanning Representations

As in most of the problems, in the floorplanning design, the representation plays a very important role. Many floorplanning representations have been proposed for 2D design, and these representations have been adapted to the 3D scenario. However, as will be presented in the following, these representations do not cover the needs for thermal aware floorplanning.

One of the most extended representations is the Combined Bucket Array first proposed in [29]. This representation encodes the layout of each layer using a transitive closure graph (TCG), and a bucket structure to encode the z-axis neighboring. The layer encoding is done depicting the geometric relations among the functional units in the same layer with two different graphs. The horizontal graph represents horizontal connections between blocks, and the width of these elements is given by its node weight. Combing the description given by the horizontal graph and it correspondence with the vertical graph a complete map of the layout of the layer can be done. TCG representation has been also used in 2D floorplanning and the extension for 3D designs has been done by including a bucking structure. For this purpose, layers are split in regions which are saved in a bucket structure which contains the functional units in those areas for all the overlapping layers. Combining TCG with the bucket structure, the whole 3D stack is characterized. Figure 4.12 depicts how a 2-layer design is described using this representation.

Other authors have preferred different representations for the 3D floorplanning description using ordered trees (O-trees) [11]. Many variations of the O-trees have been done in order to be able to represent 3D structures. The representation that will be analyzed is the one presented in [55] called Double Tree and Sequence (DTS). In order to represent a 3D design with

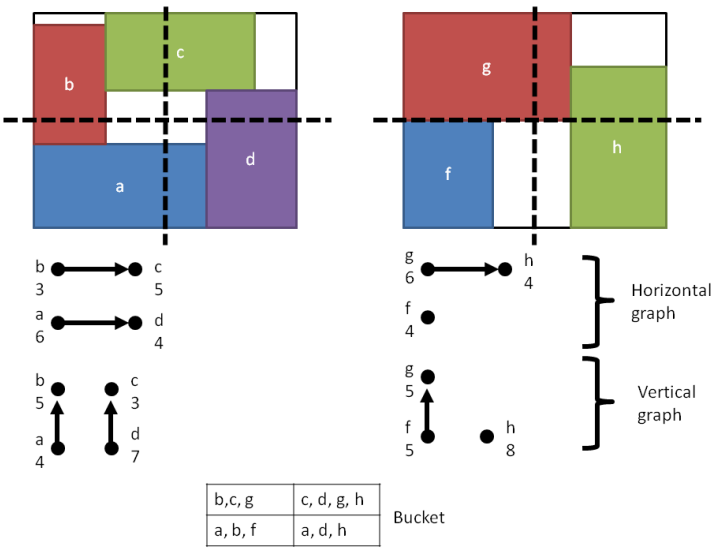


Figure 4.12: Example of a 2-layer 3D design described by the CBA representation.

this method three elements are needed. As happened with the CBA two trees are needed to represent the relationship among components in the horizontal and vertical axis, which will be called x and y-trees respectively. Finally a sequence describing the order in the vertical direction (different layers) is built and called z-order. As was previously done an example illustrating the description of our 2-layer design is depicted in Figure 4.13.

Another representation also used for floorplanning purposes is Sequence Pair (SP). In [118], Okada et Al. use the representation for thermal aware floorplanning. SP requires a unique identification of every functional unit in the design. In this way, the entire design will be described using two sequences that will be denoted as Γ^+ and Γ^- for each layer. The following procedure encodes the floorplan by a pair of the module name sequences. For each module, we draw two rectilinear curves, right-up locus and left-down locus. The right-up locus (Γ^+) of the module is initially located at the center of module and starts to move rightward. It turns its direction up and right alternately when it hits the sides of rooms, the previously drawn lines or the boundary of the chip. The locus goes until it reaches the upper-right corner. Negative sequence can be obtained similar to the positive one, but the difference is that the negative sequence is the union of the up-left locus and down-right locus. Once again, the representation is clarified with our floorplan example in Figure 4.14

Other different approaches have been taken for general floorplan optimizations such as the extended Generalized Polish Expression (GPE). GPE is

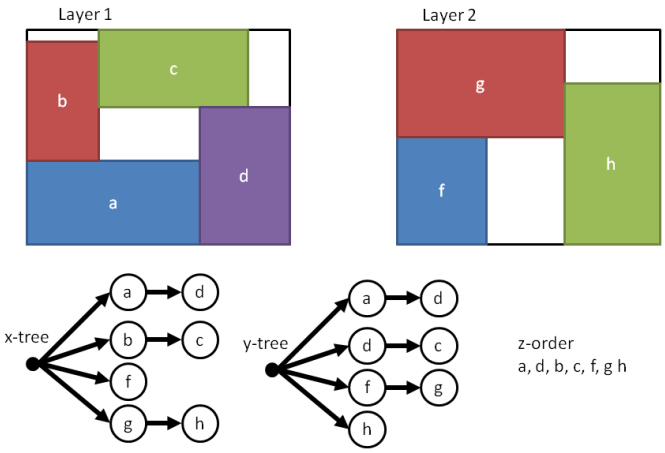


Figure 4.13: Example of a 2-layer 3D design described by the DTS representation.

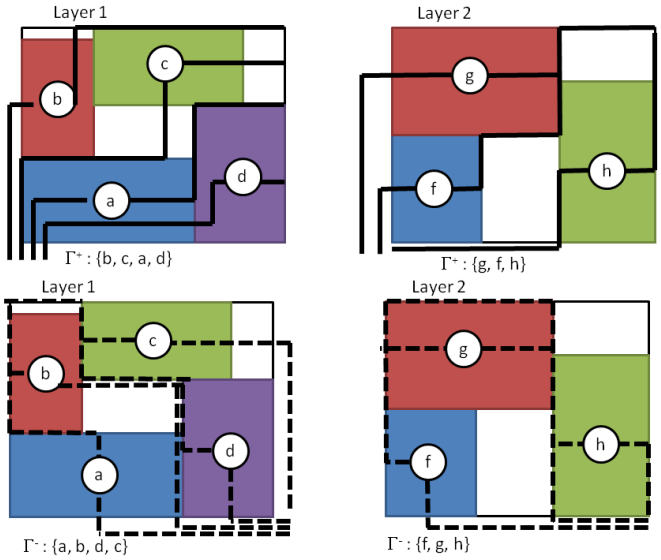


Figure 4.14: Example of a 2-layer 3D design described by the SP representation.

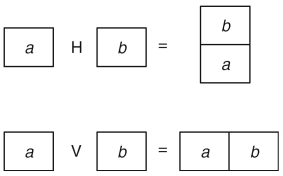


Figure 4.15: GPE Example.

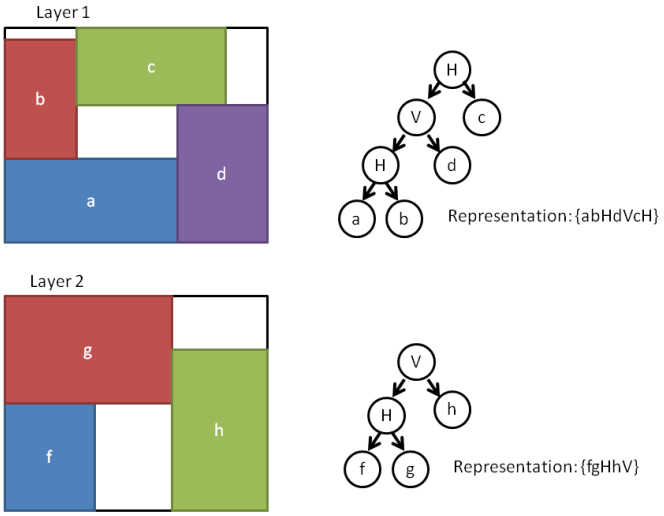


Figure 4.16: Example of a 2-layer 3D design described by the GPE representation.

a way of recording the tree in which the floorplan is sliced. GPE denotes each functional unit with an identifier and its vertical and horizontal slices with a sign. GPE is a postfix ordering of a binary tree. The slicing signs work as is shown in Figure 4.15

With this principle of the representation our 2-layer floorplan would be described using GPE as is shown in Figure 4.16.

These representations are the most commonly used among floorplanning researchers, however many other floorplan representations can be found in the literature. In this way representations such as BSG [113], Corner Block List (CBL) [71], Corner Sequence [102], Twin Binary Sequences [161], Adjacent Constraint Graph [163], etcetera. For more information about these representations the reader is referred to the proposed references.

In Table 4.2, a comparison among different representations exhibiting their solution space and its packing time is presented.

All the above presented representations can be used for thermal aware

Table 4.2: Comparing floorplanning representations. n Number of functional units. [20]

Representation	*Solution Space*	*Packing Time*
Normalized Polish Expression	$O(n!2^{3n}/n^{1.5})$	$O(n)$
Corner Block List	$O(n!2^{3n})$	$O(n)$
Twin Binary Sequence	$O(n!2^{3n})$	$O(n)$
O-tree	$O(n!2^{3n})$	$O(n)$
Corner Sequence	$\leq (n!)^2$	$O(n)$
BSG	$O(n!On^2, n)$	$O(n^2)$
TCG	$(n!)^2$	$O(n^2)$
Adjacent Constraint Graph	$O((n!)^2)$	$O(n^2)$

floorplanning, however, these representations have some drawbacks. All of them need an initial seed for the optimization engine, which can guide the problem solution towards not optimal designs. Besides, the representations are based in neighbor descriptions which makes impossible to allocate functional units surrounded by empty space which become a very useful design technique in thermal aware floorplanning.

Our representation, opposite to the ones in the literature, overcomes these problems. There is no need of defining an initial design to start the optimization flow and functional units can be placed throuhout the design volume.

4.3.1.3. Proposed Representation

As was shown in 4.1, we have that our entire 3D stack have been decomposed in small blocks. Every of those blocks i in the model $B_i(i = 1, 2, \ldots, n)$ is characterized by a width w_i, a height h_i and a length l_i.

We also have to characterize the whole design volume assigned to our final 3D stack which will have a maximum width W, maximum height H, and maximum length L. Thus, the vector (x_i, y_i, z_i) is defined as the geometrical location of functional units B_i, where the conditions that have to be fulfilled are:

$$0 \leq x_i \leq L - l_i \tag{4.28}$$

$$0 \leq y_i \leq W - w_i \tag{4.29}$$

$$0 \leq z_i \leq L - h_i \tag{4.30}$$

This set of conditions guarantee that every functional unit of the cell is inside the design volume. (x_i, y_i, z_i) are used to denote the left-bottom-back

Chromosome A	C_1	C_2	L_1	L_2	C_3	L_3	L_4	C_4
Chromosome B	L_4	L_2	L_3	C_1	L_1	C_3	C_4	C_2

Figure 4.17: Example of a chromosome from our floorplanner.

coordinate of unit B_i while we assume that the coordinate of left-bottom-back corner of the resultant Integrated Circuit (IC) is $(0, 0, 0)$.

In order to apply a MOEA correctly we need to define a genetic representation of the design space of all the possible floorplanning alternatives. Moreover, to be able to apply the NSGA-II optimization process and cover all possible inter-dependencies of the topological constraints, we must guarantee that all the chromosomes, which are the codification of our final floorplan, represent real and feasible solutions to the problem and ensure that the search space is covered in a continuous and optimal way. To this end, we use a permutation encoding [27], where every chromosome is a string of records, that represents a position in a sequence. These records, gather all the information relative to the functional unit such as the name of the functional unit or its width and length. Other information relative to the floorplan which is of vital importance for the thermal aware floorplanning as the connection map and power consumption of the functional units is stored in a separate structure that has not to be codified since it is common to every floorplan. The order in which the functional units appear in the chromosome determines their position in the final layout. As may be obvious, all the chromosomes which represent a feasible solution have to be of size n, being n the number of functional units.

However this codification presents some drawbacks that must be taken into account. If the example chromosomes used by our floorplanner depicted in Figure 4.17 are taken, it can be seen how first problem appears in Figure 4.18 after having applied crossover. This happens because both offspring have produced solutions that contain duplicated functional units, which of course means they are invalid.

To solve this problem some strategies could be implemented such as an error-checking function to remove duplicates or another crossover operator. As it may be obvious mutation would produce the same result with duplicated functional units. In this case it would be much easier if the problem is formulated using a different codifications as integers. The order of the genes of the chromosome will also reveal the order in which functional units will be placed. This additional information is what makes permutation encoding specially useful for ordering problems. However, a special crossover mechanism has to be still developed. There are many methods to ensure that crossover does not produce any duplicated genes such as Partially-Mapped Crossover [136], Order Crossover [119], Edge Recombination Crossover [114]... In our

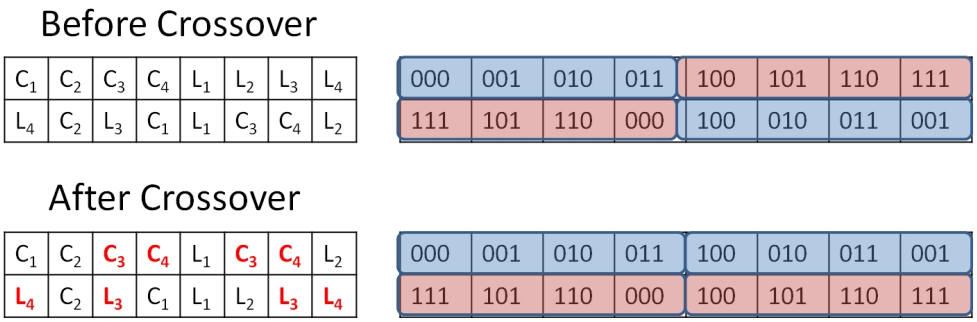


Figure 4.18: Crossover example with repeated elements.

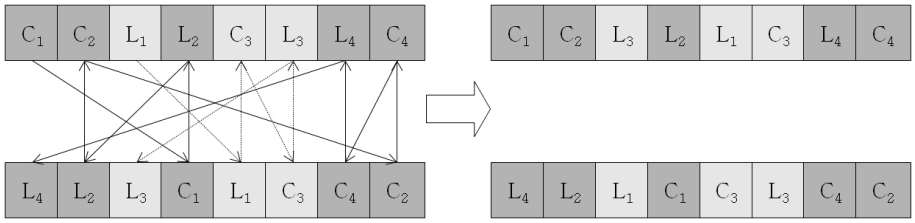


Figure 4.19: Crossover example of binary encoded chromosomes.

floorplanner we will use a different crossover technique that will be explained later.

On the other hand, mutation has to be also defined to ensure that the possible modifications in the solution do not produce any duplicated genes, verifying that any mutated chromosome is also a feasible solution to the optimization problem.

Once the importance of the encoding and the evolutionary operators has been addressed, we will explicitly show how the thermal-aware floorplanner is built.

Figure 4.17 depicts two chromosomes used by our MOEA floorplanning problem. Each chromosome in the Figure is formed by 4 cores $C_i (i = 1, 2, 3, 4)$ and 4 memories $L_i (i = 1, 2, 3, 4)$.

In every cycle of the optimization process (called generation) two chromosomes are selected by tournament. Tournament selection involves running several "tournaments" among a few individuals chosen at random from the population. The winner of each tournament (the one with the best fitness) is selected for crossover. Selection pressure is easily adjusted by changing the tournament size. If the tournament size is larger, weak individuals have a smaller chance to be selected.

Our selection operator selects two random chromosomes from the whole population and then the best of these are selected. This task is repeated twice in order to obtain two chromosomes (parents). Next, as Figure 4.19 depicts,

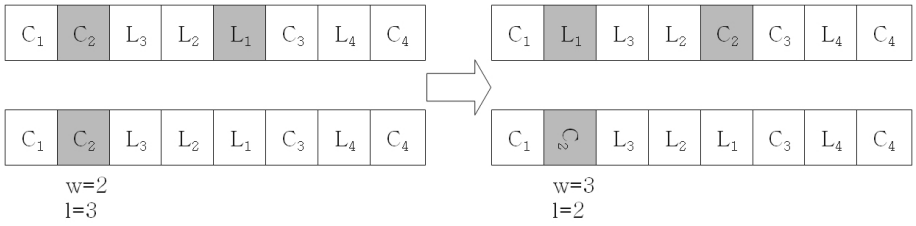


Figure 4.20: Mutation example of our thermal aware floorplanner.

cycle crossover is applied: starting with the first allele of chromosome A (C_1), then, the allele at the same position in chromosome B is taken. Next, the position with the same allele that was in chromosome B is taken in A, and add this allele to the cycle. Then, the previous step is repeated until the first allele of A is taken again which breaks the loop. Finally, the alleles of the cycle in the first child are put on the positions they have in the first parent, and take next cycle from second parent. As can be seen in Figure 4.19, dark squares are unchanged units from the parents and gray squares represent when the children are swapped. Following this process it is guaranteed that the offspring are feasible solutions to the problem avoiding the problems that may be caused by duplicity in the functional units as was seen before.

With this codification is proven that the covered space of search is given by $n!$. With these set of variables the solutions found by the floorplan are good enough for not to consider other kind of representation or crossover mechanism. The adopted representation of the chromosome and hence, the evolutionary functions designed for it, are the only representation, as far as we know, that allows the placement of functional unit in not a compacted way as Polish notation or O-tree do.

Once the crossover has created children, mutation can be executed in two different ways, both with the same probability. There are different sets of variables to be mutated. The chromosome reveals the order of the functional units and each gene of the chromosome has the information of its position (see Figure 4.20. As a result, some units are chosen and swapped as it happens with C_2 and L_1 in the first child, and others are rotated 90 degrees, as it happens in C_2 in the second child where the length (l) and width (w) of the block are swapped.

4.3.1.4. Fitness Function

Each chromosome represents the order in which the functional units are being placed in the design area. Every functional unit B_i is placed taking into account all the topological constraints, the total wire length, and the maximum temperature in the chip with respect to all the previously placed units $B_j : j < i$. In order to place a unit i, the best point (x_i, y_i, z_i) is

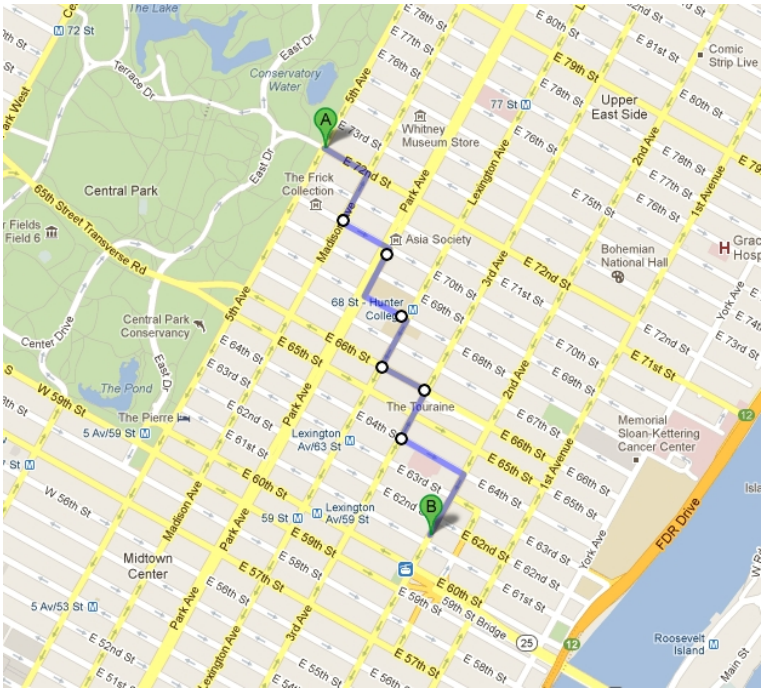


Figure 4.21: Manhattan distance between two points of the island of Manhattan.

taken in the remaining free positions. To select the best point a dominance relation is established, taking into account the following objectives in our multi-objective evaluation.

The first objective ($J1$), described in algorithm 4.3 is just related to topological relations among placed units. This objective ensures that current area created by functional units is less or equal than maximum area and also ensures no overlapping between placed units. Without this objective our floorplanner could find unfeasible solutions, that is why $J1$ is considered as the first objective to be fulfilled.

The second objective ($J2$) that our floorplanner evaluates is the wire length. Wire length is important because our objective is not only to reduce temperature in the chip, but also to keep admissible communication delays due to wire length. The wire length is approximated as the Manhattan distance between interconnected blocks. Manhattan distance is given by the sum of the absolute differences of the block coordinates. The name alludes to the city plan distribution on the island of Manhattan, where the distance between two points has to be followed by the streets. In Figure 4.21, it can be seen how the distance between two points when moving on the island, is given by the sum of all the segments of the streets. This distance between the two points $\mathbf{p}$ and $\mathbf{q}$, is formulated as:

$$d_{Manhattan}(\mathbf{p},\mathbf{q}) = \|\mathbf{p}-\mathbf{q}\|_{Manhattan} = \sum_{i=1}^{n} |p_i - q_i|, \tag{4.31}$$

This objective is set in order to minimize the distance among connected units which is made taking into account the existence of TSVs. This optimization will be described later in more detail.

The third objective ($J3$) is a measure of the thermal impact, based on the power consumption. To compute the thermal impact for every value of power consumption we cannot use inside the optimization process, an accurate thermal model, which includes non-linear and differential equations because it would make our optimizer unfeasible in terms of execution time for architectures with many cores. In a classical thermal model, the temperature of a unitary cell of the chip, depends not only on the power density dissipated by the cell, but also on the power density of its neighbors. The first factor refers to the increase of the thermal energy due to the activity of the element, while the second one is related to the diffusion process of heat [120].

As was seen in section 4.1 temperature matrix (T) that is given by equation 4.32, where P is the power matrix, has to be solved. For the steady state the equation is reduced to $T = R^{-1}P$. Matrix R presents high complexity because it is a three dimensional matrix. This makes the evaluation process slow, which means that the extended thermal model should not be considered in the evolutionary algorithm because it would make the algorithm unfeasible in terms of computation time.

$$C\frac{dT(t)}{dt} + RT(t) = P(t) \tag{4.32}$$

Instead of considering the extended model, is assumed that the hottest blocks will be placed as far as possible in the 3D stack at the end of the optimization process. It means that the main contribution to the temperature of the hottest functional units will be proportional to their own power density, mainly because neighbor functional units will be heat sinks which have a very low impact in temperature when they are compared with heat prints of the heat sources. Thus, the contribution of its neighbors does not change significantly the thermal behavior of the functional unit. This approximation is then considered in the optimization process and many calls to the thermal simulator are saved, improving the performance of the optimization loop and its execution time. Therefore, given a 3D stack composed by functional units whose power consumption is known, the temperature of the unit i can be modeled by its power density (dp_i) as described in equation 4.33:

$$T_i = \sum_{j=1}^{N} A_{i,j}^{-1} P_j \approx A_{i,i}^{-1} P_i \propto dp_i \tag{4.33}$$

By minimizing T the algorithm will try to place the hottest functional units as far as possible. However it must be noted that the previous T is not the actual temperature of the unit. In order to obtain the temperature of the design a complete thermal model have to be executed. Thus, our remaining objectives can be formulated as:

$$J_3 = \sum_{i<j \in 1..n} (dp_i \cdot dp_j)/(d_{ij}) \tag{4.34}$$

where dp_i is the power density of block i, and d_{ij} is the Euclidean distance between blocks i and j.

As can be seen now, for this objective we have used Euclidean distance. This is because heat is spread homogeneously in all directions while, our computed wire length could only be approximated by the Manhattan distance given by our decomposition grid.

4.3.2. TSV Optimization

As was previously shown in Figure 4.11, the next step of our optimization flow is to minimize the wire length in the process of placing TSVs.

The problem has been well studied by many authors such as [88], [101] or [87]. In Figure 4.22, it can be seen how the number of TSVs impacts on the wire length (WL). In the first case, the experiment has found that in the first case a minimum in WL was achieved placing 467 TSVs. If more vias are added to the stack, the wire length will be increased. When this happens the situation is the same as in the region C of Figure 4.23. On the other hand if more TSVs are added and the wire length decreases we would be moving in region A. Our problem is, how many TSVs do we have to place? and, where do we have to place them in order to find the minimum wire length and being in region B? That question will be answered by our floorplanner which will take the 3D design with the functional units already placed and will create a Pareto front with the optimum set of solutions provided the technological constraints.

Technologically,TSVs can only connect two layers. Because of technological constraints, in the fabrication process, the TSVs are made from one layer to any other one, but the connection can be made between two different inner layers, making the TSVs very useful in order to reduce wire length in the design. A schematic view of three TSVs in a 3D IC is depicted in Figure 4.24. TSV 1 connects layers 2 and 1, TSV 2 connects layers 4 and 1 and TSV 3 joins layers 3 and 2, which means that the hole in layer 1 is no longer necessary.

The routing solution taking into account the existence of TSVS in the design simplifies the wire routing and the technology processes currently available for TSV integration in 3D chips. The placement of the TSVs is

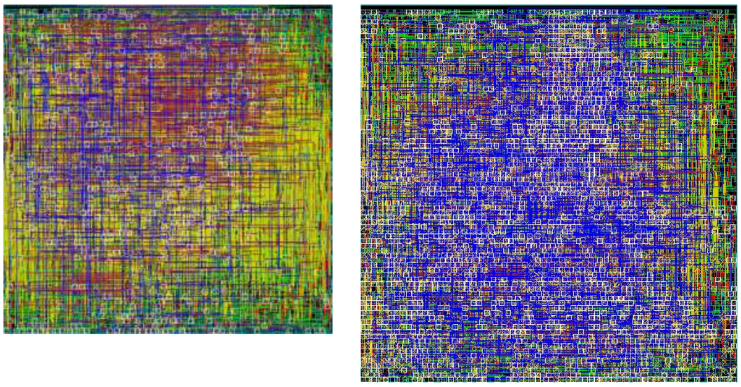
Algorithm 4.3 Floorplanning algorithm

Require: G is the number of generations. N is the population size.

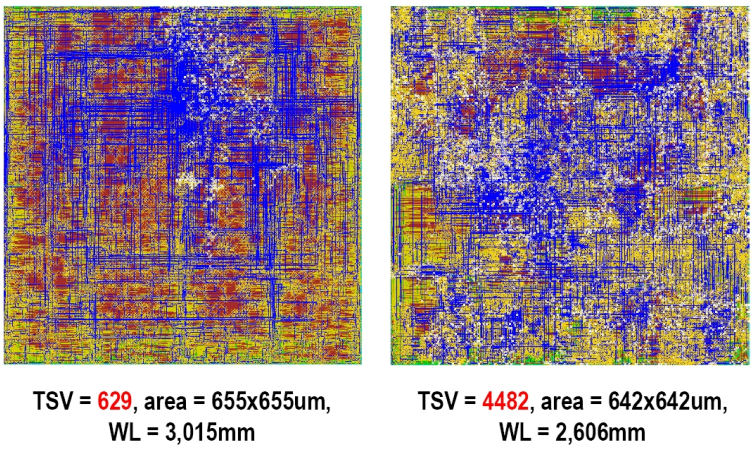
function main()
P = initialize() {P is the first random population}
evaluate(P) {P is evaluated}
for $g = 1$ **to** G **do**
  $\hat{P} = \emptyset$ {New empty population}
  for $n = 1$ **to** $N/2$ **do**
    $\hat{P_s}$ = select(P) {Select two individuals,}
    $\hat{P_c}$ = crossover($\hat{P_s}$) {perform crossover ...}
    $\hat{P_m}$ = mutation($\hat{P_c}$) {and mutation}
    $\hat{P} = \hat{P} \cup \hat{P_m}$
  end for
  evaluate($\hat{P}$)
  $P = P \cup \hat{P}$
  reduce(P) {Standard NSGA-II reduction mechanism}
end for

function evaluate(P)
for all $I \in P$ **do**
  $M \leftarrow 1$ {Matrix $L \times W \times H$ with free cells in the 3D-IC, i.e., where functional units can be placed}
  for $i = 1$ **to** n **do**
    $B_i \leftarrow I_i$ {The i-th gene in individual I (I_i) represents the i-th block (B_i) to be placed in the current 3D candidate design}
    $f_i^* \leftarrow \infty$
    for all $(x_i \in [0..L - l_i], y_i \in [0..W - w_i], z_i \in [0..H - h_i])$ **do**
      $J_1 \leftarrow$ check_topology_constraints$(x_i, y_i, z_i, i-1)$ {Number of topology constraints violated with the previous $i-1$ blocks already placed}
      $J_2 \leftarrow$ compute_wire$(x_i, y_i, z_i, i-1, M)$ {This function computes wire length according to Manhattan distances between connected blocks in the range $[1..i]$. It also asserts that a TSV can be created if two blocks are placed on different layers (it is easily computed using M)}
      $J_3 \leftarrow$ compute_temp$(x_i, y_i, z_i, i-1)$ {Compute eq. 5 with $i < j \in 1..i$}
      if B_i is a core **then**
        $f_i \leftarrow J_3$
      else
        $f_i \leftarrow J_2$
      end if
      if $f_i < f_i^*$ **then**
        $f_i^* \leftarrow f_i$
        $x_i^* \leftarrow x_i, y_i^* \leftarrow y_i, y_i^* \leftarrow y_i$
      end if
    end for
    $B_i \leftarrow (x_i^*, y_i^*, z_i^*)$ {Assign best coordinates to each block}
    update(M, B_i)
  end for
  $I \leftarrow (J_1, J_2, J_3)$ {Assign multi-objective values to each individual}
end for

optimized by another multi-objective genetic algorithm. The problem of the placement of TSVs in the remaining free cells, requires a previous analysis of



(a) 8086 microprocessor (20K gates)



(b) OpenSPARC microprocessor (300K gates)

Figure 4.22: Different TSV optimization images [101].

the free available vertical cells. The existence of enough free vertical cells has been guaranteed in the previous optimization step when all the functional units were placed.

As was done in the previous optimization loop, the first step that must be taken is to define the encoding of the chromosome that will be the solution to the problem. Opposite to the previous case, where the functional units had to be described in a single chromosome, indicating their positions within the stack, now the chromosome which defines the positions of the TSVs is much easier. An example of the chromosome used to optimize the placement of the TSVs, is depicted in Figure 4.25.

Our first MOEA has already placed the functional units in the first phase, configuring the 3D stack. However the functional units are separated by

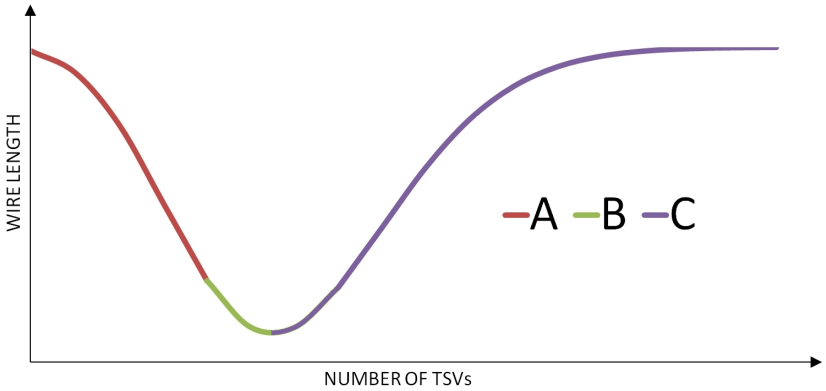


Figure 4.23: Wire Length vs Number of TSVs.

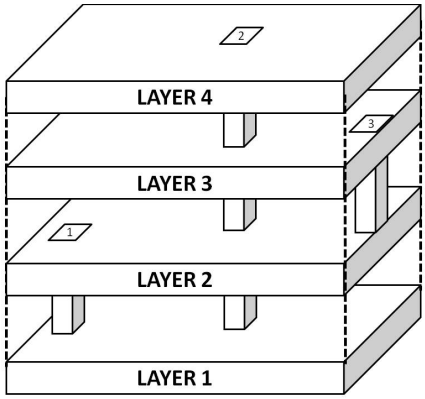


Figure 4.24: Schematic view of TSVs in a 4-layer IC.

one or more layers which make communication impossible. In order to communicate layers we have to check the resultant 3D stack to examine how the TSVs can be allocated in the design volume.

The algorithm has to check and analyze the optimized stack. With this analysis the algorithm examines the remaining free cells, where no functional units are placed. These empty cells are taken and an empty array of $x - y$ coordinates of allowed TSVs is built. Given a 3D IC with N layers, a first region of this array contains the coordinates of TSVs connecting layers *Top* and 1, a second region contains the coordinates of TSVs connecting layers *Top* and 2, and so forth, as shown in Figure 4.25. If the total number of allowed TSVs is M, next a chromosome of M 0-1 variables is built. Variable M is set to delimit the space of search of the algorithm, and taking into account that communication bandwidth requirements have to be fulfilled.

If the gene of the chromosome is 1, a TSV is inserted in the corresponding (x,y) position and it connects the number of layers defined in the corresponding region. In this way, Figure 4.25 encodes 7 TSVs in four layers

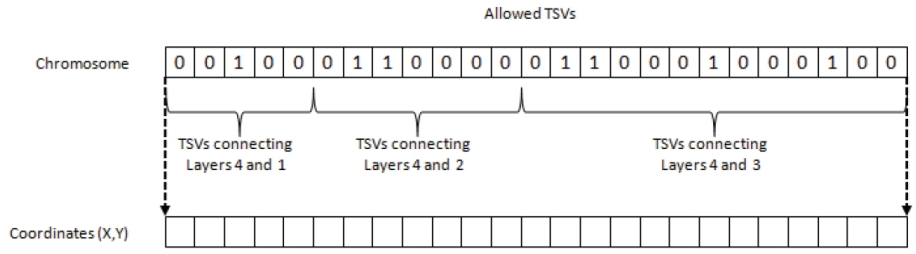


Figure 4.25: Chromosome for the optimization of TSVs.

($N = 4$):

- 1 TSV connecting layers 4 and 1
- 2 TSVs connecting layers 4 and 2
- 4 TSVs connecting layers 4 and 3

The corresponding (x,y) coordinates are stored in a separated array of coordinates. Despite the first optimization algorithm is able to reserve enough cells to ensure communication requirements, it can also place wires at the edge of the chip using wire bonding technology.

Using this representation, a Non-dominated Sorting Genetic Algorithm II (NSGA-II) is executed,following the same strategy used for the placement of functional units.

Using the above presented chromosome as an input, and the maximum number of TSVs, the algorithm returns a set of solutions, considering the number of TSVs and the total wire length. With these two variables a Pareto front of solutions is built. These solutions are found in region B of Figure 4.23.

Although the algorithm returns a set of feasible optimized solutions the designer has to choose the solution for the IC design among all the possible solutions given by the Pareto front, taking into account the benefits that including more and more TSVs have in the design versus the economic cost in which the design is incurred. A trade-off between costs and benefits must be reached.

The optimization algorithm considers that a minimum number of TSVs must be included in the design in order to fulfill communication constraints. The minimum number of TSVs is calculated considering the communication bandwidth among cores. The maximum number of TSVs is given by the technological parameters of the TSVs and the amount of data that is transferred [51]. From this study we can consider that the TSVs behave like local wires, which means, that the lumped model is accurate enough to estimate the delay produced by the TSVs and hence the bandwidth. The bandwidth

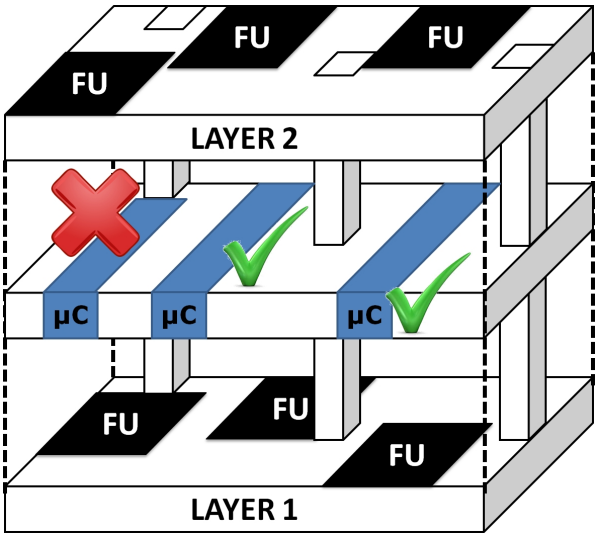


Figure 4.26: Example of channel routing.

(BW) of N numbers of TSV (N_{TSV}) in any given structure can be estimated by measuring the rate of bit streaming of each TSV multiplied by the total number of TSVs in the bundle. Thus, the bandwidth will then be expressed as:

$$BW_{max} = \frac{N_{TSV}}{T_{d,RC}} \tag{4.35}$$

where $T_{d,RC}$ is the delay introduced by a TSV and, it is function of the diameter (d) and the resistance-capacitance values (RC) that are calculated using the lumped model.

4.3.3. μchannel Optimization

Following the optimization flow presented in Figure 4.11, the next step of our incremental floorplanner is to optimize the placement and the number of μchannels.

Once the 3D IC has been thermally optimized by our proposed floorplanner, having placed the functional units and the TSVs, we run the thermal model described in Section 4.1, to evaluate the temperatures in the chip, calculating the temperature matrix T for every cell in the stack.

The information provided by the analysis of the temperatures of the stack is used as an input to the μchannel optimizer. Additional information of the stack design, including the description of the positions of the functional units and over all, the position of TSVs is also used as an input to our proposed liquid channel optimizer.

The optimization is carried out by a new evolutionary algorithm, whose evaluation module is presented in Algorithm 4.4. Since liquid channels flow right above functional units, the only topological constraint that exists in the design is TSV mapping. As can be seen in Figure 4.26, μchannels have to avoid TSVs.

As it was done for the placement of TSVs, a set of available coordinates is built. As μchannels are supposed to flow from south to north the coordinates that the algorithm will have to assign to every μchannel are now x and z, which means that a chromosome of $x - z$ coordinates is built.

Algorithm 4.4 Microchannels algorithm

Require: I is the current individual to be evaluated. T is the set of temperatures obtained with the thermal model.

function evaluate(I)
$J_1 \leftarrow \sum_{i=1..N} I_i$ {Number of liquid channels.}
$J_2 \leftarrow 0$
$\hat{T} \leftarrow T$
for all $I_i \in I$ **do**
  if $I_i = 1$ **then**
    $(x_i, z_i) \leftarrow I_i$ {Every gene is referred to a concrete (x_i, z_i), where z_i is the current layer for the i-th channel and x_i its x coordinate}
    for $y_i = 0$ **to** $W - 1$ **do**
      $\hat{T}(x_i, y_i, z_i) = 342.46ln(\hat{T}(x_i, y_i, z_i)) - 1664.4$
      $\hat{T}(x_i - 1, y_i, z_i) = 321.28ln(\hat{T}(x_i - 1, y_i, z_i)) - 1541.5$
      $\hat{T}(x_i - 2, y_i, z_i) = 293.60ln(\hat{T}(x_i - 2, y_i, z_i)) - 1380.8$
      $\hat{T}(x_i + 1, y_i, z_i) = 321.28ln(\hat{T}(x_i + 1, y_i, z_i)) - 1541.5$
      $\hat{T}(x_i + 2, y_i, z_i) = 293.60ln(\hat{T}(x_i + 2, y_i, z_i)) - 1380.8$
    end for
  end if
end for
$J_2 = \sum_{x_i, y_i, z_i} \hat{T}((x_i, y_i, z_i))$
$I \leftarrow (J_1, J_2)$ {Assign multi-objective values to this individual}

The first objective J_1 to be evaluated now, examines the number of μchannels deployed in the stack. The restriction of maximum number of channels is made by the designer. The more μchannels are added to the stack the better temperature behavior will be observed. Since μchannels entail an additional power consumption the designer will have to find the trade off between additional technological costs to create the μchannels, the extra power consumed by the pumping system and the thermal improvement.

The second objective is the new set of temperatures J_2, estimated using the effect that a μchannel has over the surrounding area.

The design of μchannel implies not only fabrication cost but also external power consumption, hence the importance of optimizing both, the number of channels in the stack and their placement to decrease the temperature as much as possible. Some studies as [31] have used an homogeneous distribution of the channels achieving good thermal results, however similar

temperatures are reached with less channels if they are placed at certain optimized positions. But in order to be able to optimize the positions of the μchannels in the 3D stack, some previous studies must be done.

The evaluation function of the MOEA takes into account the cooling effect that a liquid channel has. In order to evaluate this cooling effect, several thermal simulations were conducted. In these simulations the cooling effect that a μchannel has over the immediate bottom cell and its two first neighbors is evaluated. This reduction in the temperature follows a logarithmic trend with temperature that can be seen in figure 4.27. Introducing this data in the evaluation of our proposed evolutionary algorithm, we can find optimized solutions to the positions of the μchannels.

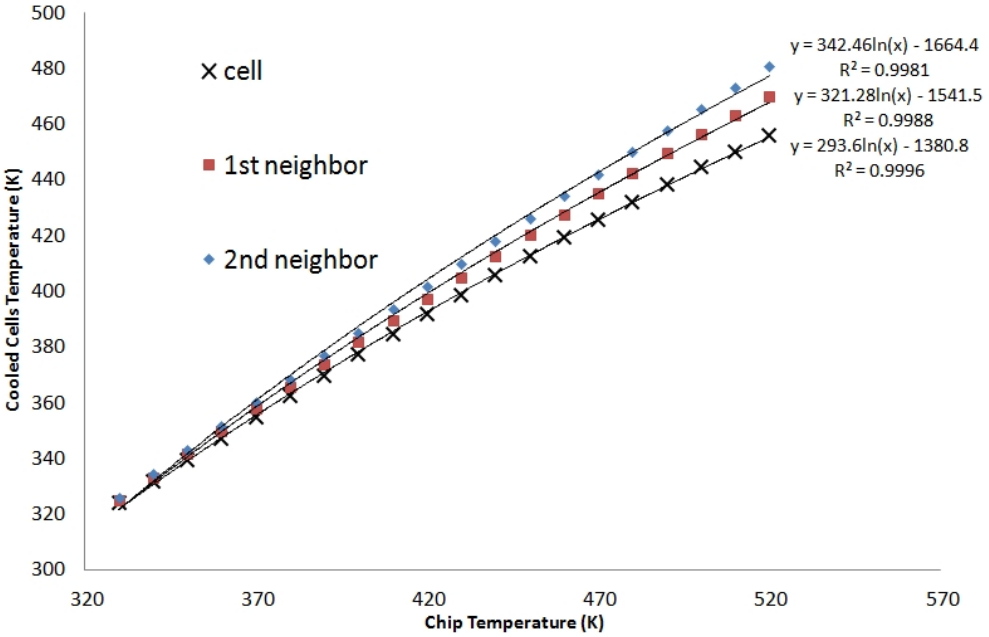


Figure 4.27: Regression for evaluation function in liquid channels optimization.

Our proposed algorithm returns a set of solutions for a maximum number of liquid channels N. This makes a Pareto front approximation, and again as it happened with the distributions of TSVs, the designer, considering economic costs and thermal benefits, will choose the best configuration for each scenario.

4.3.4. Air Channels and Floorplan Restrictions

Up to this point it has been explained how the floorplanner is able to thermally optimize a given initial design, first placing the functional units taking into account the power dissipated and, the TSVs to minimize the

impact of the wire length overhead, produced when the functional units are spread throughout the stack. Then, to alleviate the problem of warmer regions, μchannels are placed in certain positions optimizing both, the number of deployed μchannels and the temperature distribution in the 3D IC.

However, the floorplanner has to cope with some restrictions that were not contemplated before. Creating thermal domains in a chip is one of the major contributions of this PhD thesis as explained in 3.2. Warm regions are a revolutionary method to keep heat concentrated in certain areas preventing, with air isolation channels, heat spread from hot to cold regions. This could seem awkward, but if this approach is combined with liquid channels, better results are obtained.

When the chip is isolated by air channels the optimization placement process is guided. This process works like the one described before but restricting certain areas of the chip to chosen functional units, according to their power consumption.

Hot areas would be composed by functional units that have a high power density, on the other hand, elements with a lower power consumption will be placed, all together, in cold regions, as can be seen in Figure 3.14. In the figure, functional units with big power consumption as cores (C) are placed in warm regions, while colder functional units with a low power consumption as memories (M), occupy other areas. This is specially useful if more layers come into play. Stacking several layers with warm and cold regions vertically makes the achieved thermal pattern to be more homogeneous.

The regions are topologically designed before the optimization phase for every layer of the stack. The description file which contains all the information about the topological distribution of the 3D IC, and the information about the functional units, their size and power consumption, are the inputs for the optimization algorithm. The constraints in the design are analyzed by the floorplanner and the constraints are evaluated in order to guide the placement of functional units and TSVs in the first step, and then, the placement of μchannels.

The considered algorithm with the constraints in the design works exactly in the same way as the one described in algorithm 4.3, but another topological restriction is included in the process, obviously neither functional units nor TSVs can occupy cells previously designed as air channels.

As will be shown in next Chapter, air channels improve not only the thermal distribution throughout the IC but also the technological cost and optimization time, since the algorithm examines an enclosed space of search.

4.4. Description of Source Code Files

All the optimization and calculation system described in the above sections are programmed in different steps and platforms. The complete opti-

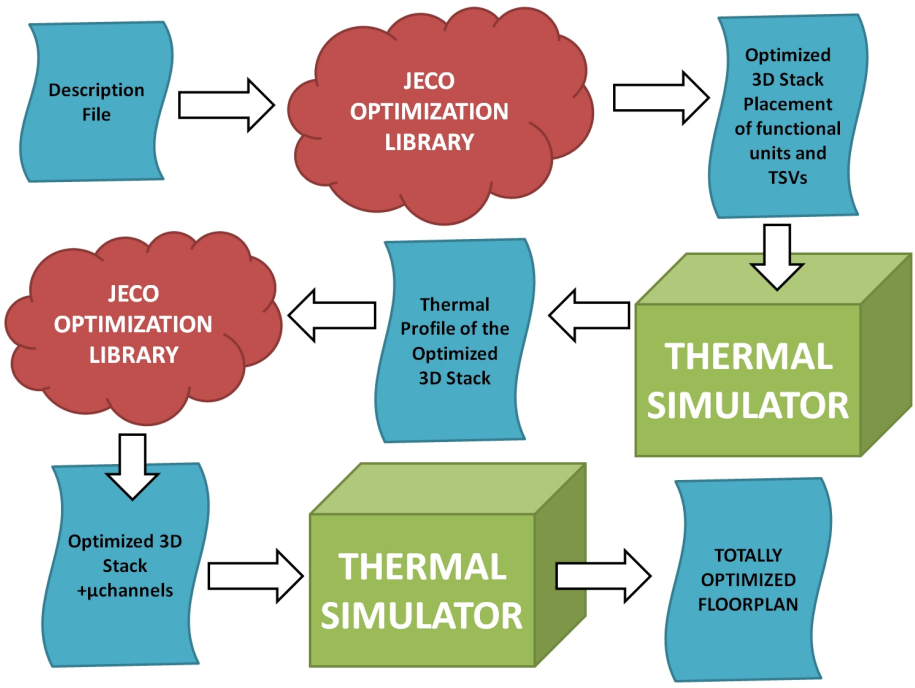


Figure 4.28: Optimization platform schema.

mization flow is presented in Figure 4.28.

As can be seen in Figure 4.11, the input to the optimization process is the description file which contains a list of the different functional units as well as geometrical parameters. The description file has to include the most important information about the technological variables such as, width and length of the stack, and the total number of layers of the 3D IC. Since the stack is divided into small cells, the grid has to be also described in this file specifying the size of the cell. If the design volume had air isolation channels mapped, these would be also contained in the initial description file.

Next, the description file has to include information about the functional units, indicating their location, name and size, and also the power consumption of the unit. Since the optimization algorithm for allocating the functional units does not need any initial feasible floorplan as a seed, the positions of the functional units are not defined. The best way to describe the information is using a Extensible Markup Language (XML) file which is the standard reference file for data interchange. The schema used for the floorplanning description is seen in Figure 4.29. The file represents a main element *Floorplan* with attributes which describe the general settings of the 3D IC. The floorplan has five children:

1. *Blocks*: It describes the functional units that will conform the IC, specifying the location of the unit, its shape and its power density

consumption.

2. *Couplings*: It describes the connections among units. This child is mainly used to decrease the wire length overhead and to route TSVs.

3. *ThermalVias*: It contains the position and the initial and ending layer of every TSV in the design.

4. *LiquidChannels*: It sets the position of the channel and the layer in which it is placed.

5. *Temperatures*: This final child specifies the temperature of every cell of the stack.

The first two elements are set from the beginning of the problem, with a seed floorplan, and the rest of elements are empty. The XML file with the description of the initial floorplan is used as an input to the optimization algorithm using Java Evolutionary Computation (JECO) library [127]. JECO is a Evolutionary Computation Library developed in Java. It includes a variety of evolutionary optimization techniques such as genetic algorithm, genetic programming, evolutionary mapping methods, particle swarm optimization, ant colonies, etc. The algorithm developed using JECO, calculates the optimum positions of the functional units and TSVs as described in Section 4.3.

The output to this first optimization is another XML file with the description of the optimized floorplan including also the information about TSVs location. The optimized floorplan is used as an input to the thermal simulator. The simulator is programmed following the equations and guidelines described in Section 4.1 using Matlab©. The thermal simulator returns the temperature values for every cell of the divided stack and writes it in the XML file. This file will again be used as an input to the JECO optimizer, but this time the optimization library will search the best position to place the μchannels following the guidelines described in Section 4.3.3. The optimization process gives as an output, a modified file which includes the characteristics of the liquid channels which are their position and the number of μchannels deployed in the stack.

```
<?xml version="1.0" encoding="UTF-8" ?>
<xsd:schema
xmlns:xsd="hhtp:www.w3.org/2001/XMLSchema">
<xsd: element name="Floorplan">
    <xsd: attribute name="Version" type="xsd:double"/>
    <xsd: attribute name="CellSize" type="xsd:double"/>
    <xsd: attribute name="Length" type="xsd:double"/>
    <xsd: attribute name="Width" type="xsd:double"/>
    <xsd: attribute name="NumLayers" type="xsd:double"/>
    <xsd: attribute name="NumPowerProfiles" type="xsd:double"/>
        <xsd: element name="Blocks" type="xsd:string">
            <xsd: attribute name="id" type="xsd:int"/>
            <xsd: attribute name="name" type="xsd:string"/>
            <xsd: attribute name="type" type="xsd:int"/>
            <xsd: attribute name="xMin" type="xsd:int"/>
            <xsd: attribute name="xMax" type="xsd:int"/>
            <xsd: attribute name="yMin" type="xsd:int"/>
            <xsd: attribute name="yMax" type="xsd:int"/>
            <xsd: attribute name="zMin" type="xsd:int"/>
            <xsd: attribute name="zMax" type="xsd:int"/>
            <xsd: attribute name="x" type="xsd:int"/>
            <xsd: attribute name="y" type="xsd:int"/>
            <xsd: attribute name="z" type="xsd:int"/>
            <xsd: attribute name="l" type="xsd:int"/>
            <xsd: attribute name="w" type="xsd:int"/>
            <xsd: attribute name="h" type="xsd:int"/>
            <xsd: attribute name="dp" type="xsd:double"/>
        </xsd: element>
        <xsd: element name="Couplings" type="xsd:string">
            <xsd: attribute name="idFrom" type="xsd:int"/>
            <xsd: attribute name="idTo" type="xsd:int"/>
        </xsd: element>
        <xsd: element name="ThermalVias" type="xsd:string">
            <xsd: attribute name="zIni" type="xsd:int"/>
            <xsd: attribute name="zEnd" type="xsd:int"/>
            <xsd: attribute name="x" type="xsd:int"/>
            <xsd: attribute name="y" type="xsd:int"/>
        </xsd: element>
        <xsd: element name="LiquidChannels" type="xsd:string">
            <xsd: attribute name="id" type="xsd:int"/>
            <xsd: attribute name="x" type="xsd:int"/>
            <xsd: attribute name="z" type="xsd:int"/>
        </xsd: element>
        <xsd: element name="Temperatures" type="xsd:string">
            <xsd: attribute name="Value" type="xsd:double"/>
            <xsd: attribute name="x" type="xsd:int"/>
            <xsd: attribute name="y" type="xsd:int"/>
            <xsd: attribute name="z" type="xsd:int"/>
        </xsd: element>
</xsd: element>
</xsd: schema>
```

Figure 4.29: xsd schema for the floorplanning description.

Chapter 5

Experimental Set-Up and Results

Quality is never an accident; it is always the result of intelligent effort.

John Ruskin

In the previous chapters, it has been presented the thermal problem of the 3D integration, and the different techniques that can be found in the literature to cope with the temperature issues. It has also been explained the developed techniques proposed by this PhD study, and the theoretical basis of the methods which include the optimization in the design phase of the stack and the verification by the use of an accurate thermal simulator.

In this chapter it will be first shown the experimental set up which is used to show how the optimization platform works. We will start with simple and academic examples to later move to the realistic scenario used to illustrate the benefits exhibited by the optimization process.

5.1. 3D Thermal Aware Floorplanning Guidelines

The use of simple benchmarks is extremely important in the first phases of the development of the platform. The initial study of simple benchmarks gives the guidelines for the optimization design, contributing with design rules or clues for the optimization flow.

From the work initiated in [35], some design guidelines were extracted, depending on the final purpose of the 3D stack. In this work cores and memories were placed in certain positions to obtain general conclusions on how heat is transferred through the chip and among layers.

In this simple and first study, only three cores are deployed in each layer and their power trace is characterized by an emulation platform as the one

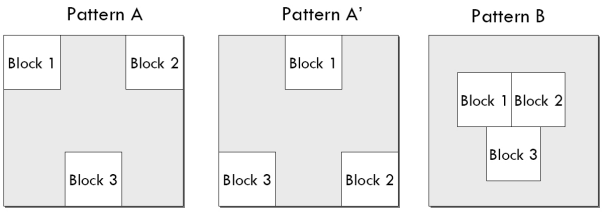


Figure 5.1: Intra-layer distribution.

presented in [7].

In order to inspect the consequences of the placement of the units two different variables have to be studied because both, the intra-layer distribution, and the inter-layer placement affect the thermal behavior of the stack.

In figure 5.1 the first two patterns (Patterns A and A') are shown. In this distibution the blocks are kept as much distant from each other as possible (and are then placed on the chip boundary). Pattern A' is just Pattern A flipped vertically, in order to stack two similar patterns dividing vertically place one block over another.

In the third pattern (Pattern B), the blocks are kept as close as possible, and are placed in the center of the layer.

In these patterns, the empty space is dedicated to routing and communication units. Since this work is focused on the thermal impact of the distribution, these communication units are not considered in the description for their negligible thermal impact.

The two patterns represent two configurations optimized for different metrics. Patterns A and A' can be thought as *thermal-driven* options, since the cooling capabilities are maximized thanks to the thermal diffusion towards the environment. Conversely, Pattern B can be seen as a *performance-driven* pattern, since keeping blocks closer will reduce the inter-block communication delay.

On the other hand, inter layer placement is relative to the distribution of cores and memories over the layers. Two abstract patterns are envisioned here. The first one is a *homogeneous* pattern, in which one layer accommodates blocks of the same type: all-memories, or all-cores. The second pattern is *non-homogeneous*, that is, it allows to mix memories and cores in the same layer.

In the homogeneous case (cases AA and BB in Figure 5.2), the number of cores clearly implies the overall floorplan, and there is then only one possible option. Conversely, in the non-homogeneous pattern (C1 and C2 of Figure 5.2), several options can be arranged, depending on whether one or two cores are placed on a given layer (three cores per layer will correspond to the homogeneous case).

Combining the different options a set of test scenarios as shown in Figure

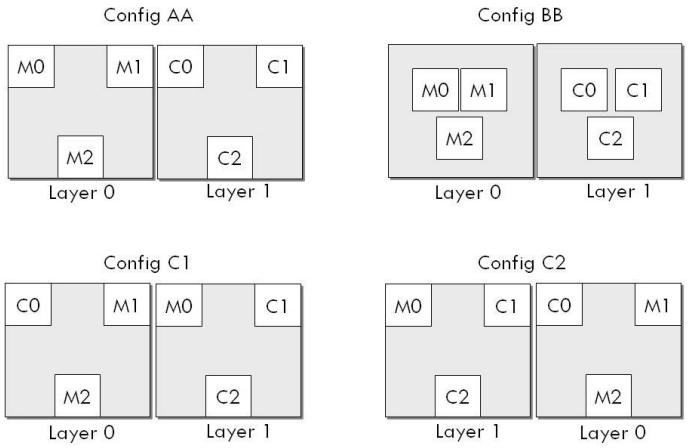


Figure 5.2: Inter layer distribution for the homogeneous and heterogeneous case.

5.3 can be designed. The configurations with nine cores are slightly different since we cannot use only patterns A (or A') and B, having nine cores and nine memories to place. Since we are constrained to the three-cores-per-layer pattern, we place all nine memory block on Layer 0 (pattern M) while the nine cores will be distributed over the three remaining layers using the basic 3-block patterns. Configuration $MAAA$ ($MBBB$) means the the nine cores are distributed according to the A (B) pattern; $MAA'A$ is like $MAAA$, with the middle layer (Layer 2) flipped with respect to Layers 1 and 3. Running the thermal simulator with the inputs of these scenarios some design guidelines can be derived form the results.

The metrics considered for the analysis of the experimental results are the interconnect delay as a function of temperature and distance, average temperature, maximum temperature and the thermal gradient. These metrics are usually found in every thermal-related analysis. The results have been collected for an execution of the application of three minutes of real time.

From the results presented in Figure 5.4 we can extract some conclusions. It can be observed that concentrating cores in the same layer (i.e., according to homogeneous patterns) will increase the gradients and the maximum temperature of the layer. This can be explained by the fact that the lateral diffusion is not able to cool down these hot spots in the chip, even if the heat sources are surrounded by heat sinks like memories. Also, it has to be noticed that those middle layers in the 3D stack present worse cooling capabilities because of the distance to the heat sink and the presence of hot spots in the upper and lower layers. Therefore, these layers show higher thermal gradients and maximum temperatures. This effect is especially exposed in layer number 3, where the heat diffused by layers 2 and 4 increases the

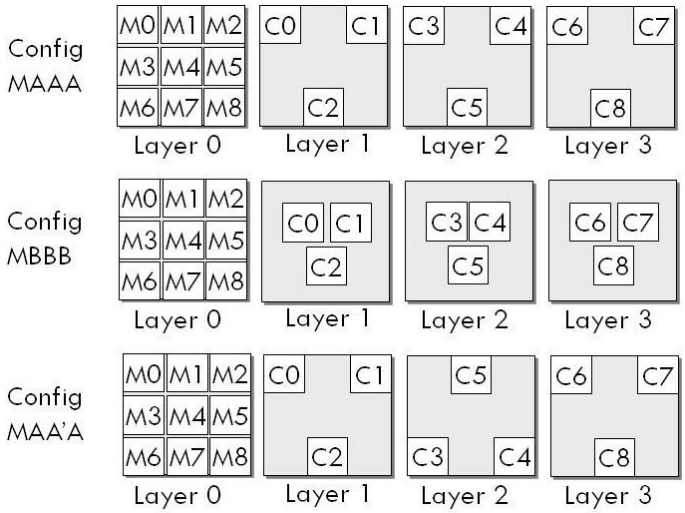


Figure 5.3: Basic test scenarios.

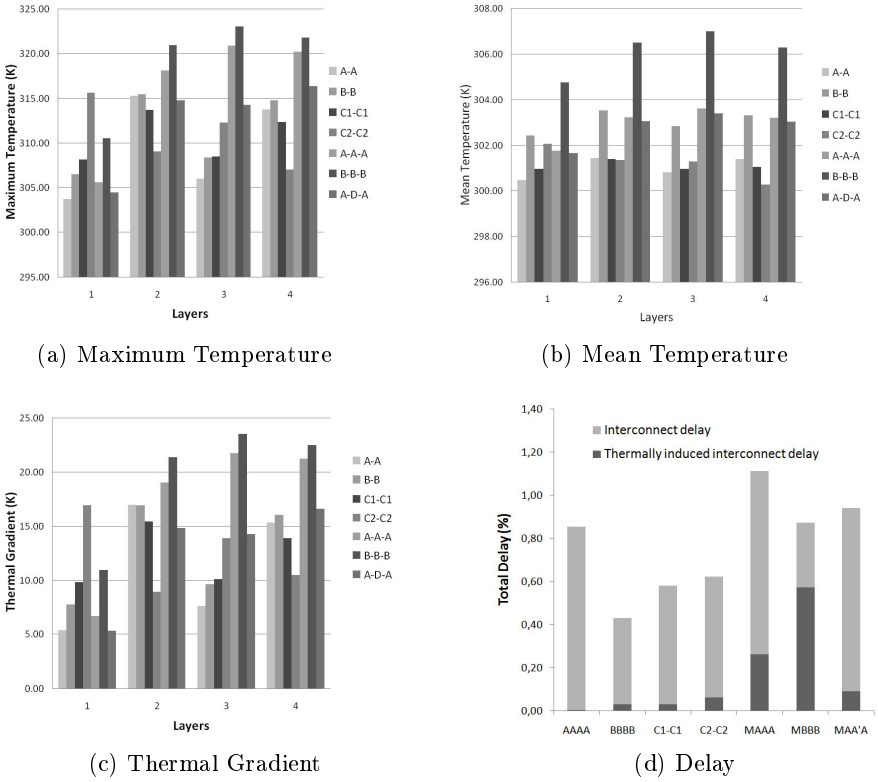
temperature of the layer above the values reported for layer number 1.

Significant variations can be observed for the average temperature of the layers. While those layers with more cores present a higher average temperature, the effect of air cooling on the top layer and the effect of the adiabatic bottom surface are clearly noticeable. The top layer, when compared with layer number 2, shows a lower average temperature due to its improved cooling capabilities. On the contrary, the average temperature is increased in the bottom layer because it is not able to diffuse the heat across the adiabatic PCB.

As it can be observed in Figure 5.4d, most part of the interconnect delay is due to the core placement and the resulting length of the communication buses, and the effect of the temperature is minimum in this, however, when the number of cores is risen up to 9, this effect is far from been negligible is dominant as in the case of $MBBB$.

With the analyzed information from the thermal simulator we can now define some working scenarios that allow us to classify the various floorplan configurations according to their transient thermal and performance behavior. This experimental work proves the need of incorporating the thermal emulation in the design flow of multiprocessor systems and, in particular, in the floorplanning of them, setting the conditions and the basic guidelines to optimize the design in 3D stacks. According to the needs in the design the results have been classified in three different application scenarios:

- **Scenario 1: Low temperature**: In this scenario, a minimum of temperature is targeted as a working condition. This temperature to minimize is calculated as the average of the mean temperatures of the



(a) Maximum Temperature

(b) Mean Temperature

(c) Thermal Gradient

(d) Delay

Figure 5.4: Basic scenario results.

layers. These working conditions are easily found in portable electronics, such as mobile telephones, where the thermal dissipation has to be reduced if the cooling capabilities of the device are restricted. Therefore, the floorplans selected for this category try to improve the cooling possibilities of the cores by spreading them across the chip surface and trying to set the heat sinks close to them (pattern A).

- **Scenario 2: High performance**: This scenario, usually found in systems with high computation requirements, selects a high-performance response instead of an efficient thermal behavior. The improvement of the performance capabilities can be achieved by reducing the communication delays among the cores. Therefore, those floorplans with a localized placement of the cores (patterns A and B) will exhibit a lower delay and higher performance. However, the thermal issues that appear in these cases have to be traded-off carefully.

- **Scenario 3: High reliability**: This scenario will focus on decreasing the thermal gradients and the maximum temperature, metrics with a high influence on the reliability of the circuits. As can be derived

after analyzing the experimental results, spreading cores in neighboring layers (distributions C_1 and C_2) is the best solution as long as this creates heat source-sink pairs. High reliability systems are commonly found in those environments where the safety and the cost are the main constraints.

With this initial and basic study of how heat is transferred in a 3D stack, we have investigated the effect that the thermal-aware floorplanning techniques can have on the thermal profile of complex 3D multi-processor systems.

With the understanding of how the placement of the functional units affect not only the other units placed in the same layers, but also the ones placed in the higher or lower, the optimizer can find good solutions in less time, because the initial searching space can be reduced to not optimal but yet good solutions designed with the "common sense" guidelines extracted from this work.

5.2. Original Floorplan.

In the previous section, several guidelines have been shown. These design conditions must be taken into account by our floorplanner in the optimization phase. However, in order to test the solutions obtained by the optimizer the target architecture has to be first defined.

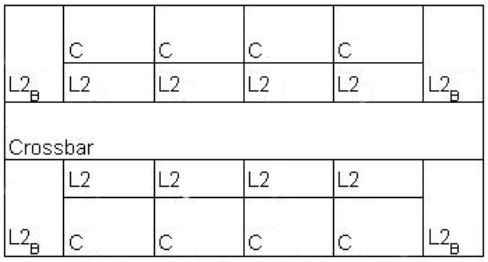
In our study two different realistic architectures have been chosen to conduct the 3D thermal aware design, Niagara 2, which is composed by 8 cores and Niagara 3 which has 16. The design is replicated in different layers in order to build a complete 3D stack.

However the original design has been slightly modified in order to include more number of cores keeping the original architecture invariant. Since our floorplanner can place a variable number of cores in every layer, the power consumption of the crossbar is scaled accordingly to the number of cores found in every layer and their required bandwidth. The inter-layer communication is resolved with a set of TSVs that route the communication signals from one layer to another.

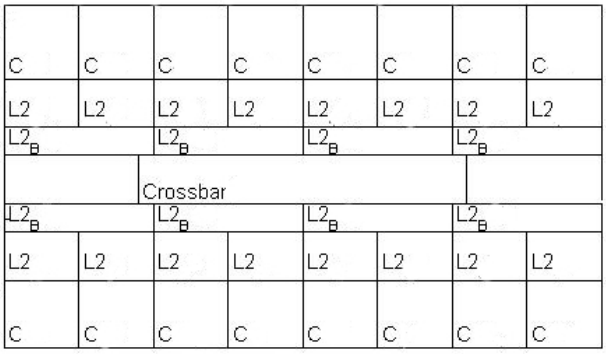
Worst case scenario has been set for power consumption. In our two realistic floorplans, power consumption is set to 84W and 139W for Niagara2 and Niagara3 respectively [1].

As has been shown in Section 4.3, the floorplanner will place the functional units that compose the 3D multi-processor architecture minimizing temperature parameters. The area is set from the beginning of the optimization, and it is the original distribution the one that sets the area of the optimization. The thermal results obtained by our floorplanner will be compared with the stacks composed by the two original layers, based on Niagara

2 and Niagara 3, whose simplified designs are presented in Figure 5.5. The C blocks are cores, $L2$ are the memories of each core, while $L2_B$ are the common memories. These two designs are disposed differently in the chip in order to build complete 3D systems.



(a) Niagara 2 floorplan



(b) Niagara 3 floorplan

Figure 5.5: Original Floorplans.

Our experimental work will be focused on the analysis of the thermal optimization achieved by the floorplanner and the additional temperature reduction by the liquid cooling system.

5.3. Experimental Set Up.

The set up that will be described in the following will be used in all the optimization, simulations and test processes. From the Niagara designs the optimization engine will take the functional units that will compose the final 3D stack. The different designs that will be optimized are built combining both Niagara 2 and 3 designs in several layers.

As it was previously said, the worst case thermal scenario is chosen. If

the worst case scenario is set, then the critical situation, in which the system has the highest temperature, will be analyzed. Any other situations with a lower power consumption will then be more thermally advantageous.

All the placement optimizations have been run in an Intel Core 2 Quad processor Q8300 system with 4 GB of RAM, under Windows 7. Liquid channel optimization and thermal analysis have been done in an Intel Core 2 Duo T6400 @2GHz and 4GB of RAM.

In the following the configuration parameters for the optimizer and the simulator will be described.

5.3.1. Optimizer Set Up

The developed algorithms that were described in Section 4.3 include different evolutionary functions that need to be parametrized and configured.

The initial population that will make the first set of possible solutions to the problem for all the presented optimization processes is chosen randomly allowing the entire range of possible solutions (the search space). The optimal solutions can not been predicted because they are not enclosed by a certain area of the space, thence, the initial population has to be spread throughout the entire search space which is achieved by creating a random initial population.

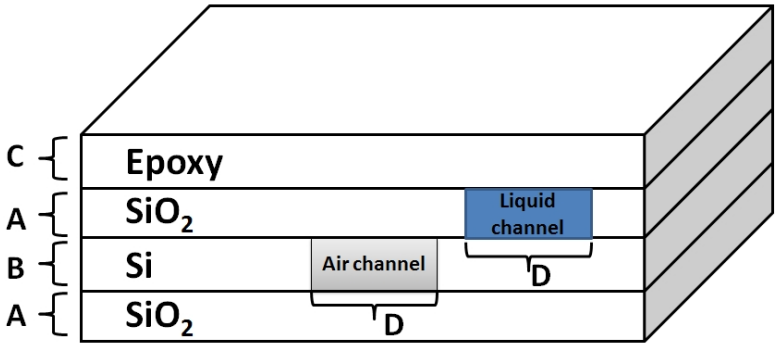
For the initial placement algorithm the population is set to 100 individuals, which has been proven to be a sufficient population to find feasible solutions. The algorithm is also configured using a probability for the crossover of 0.90. The probability for the mutations is set to 1/number of blocks. These parameters have been set using the guidelines recommended in [44].

On the other hand the algorithm which is in charge of optimize the location of the liquid channels is configured with maximum population of one hundred individuals, and a maximum number of 250 generations. The probability of mutation is set depending on the number of variables; in this particular case, it is the inverse of the number of available points in the 3D design. Then, we set a single point crossover with a probability of 0.9 and the tournament selection method, following the guidelines given in [44].

The parameters used by the algorithms are summarized in Table 5.1.

Table 5.1: Parametrization of the optimization algorithms.

	Functional Units and TSV optimization	*Liquid Channels optimization*
Population Size	100	100
Max. generations	$100 \cdot Number_{blocks}$	250
Crossover probability	0.9	0.9
Mutation probability	$1/number_{blocks}$	$1/number_{freeCells}$



A	B	C	D
50μm	150μm	25μm	300μm

Figure 5.6: Technological values of a 3D tier.

5.3.2. Simulator Set Up

The thermal model described in Section 4.1 has to be implemented in a computation environment in order to calculate the thermal profile of the solutions provided by the optimization algorithms developed in our floorplanner.

The development of the thermal model has been done using Matlab ©due to its good performance using big set of numbers and its out of the box implemented tools for calculation and viewers.

The simulator uses the XML floorplan description file provided by the functional unit placement optimizer as was previously depicted in Figure 4.28. The description file includes the whole information of the functional units such as the position of the block, the power consumption and their length and width. The simulator takes all this information and computes, following the thermal model equations, the temperatures for each of the cells in which the 3D stack is decomposed.

Each tier of the stack is physically characterized as depicted in Figure 5.6. Following the thermal model description the entire 3D stack is decomposed in small cells whose width and length is 300μm. The ambient temperature is set to 300K, the same temperature supposed for the liquid coolant that is pumped into the chip through the liquid μchannels. The flow rate for each channel is set to 0.1 liter per minute. This flow has been set in order to avoid the turbulent regime of the fluid taking into account the size of the μchannel and the physical properties of the fluid.

In Table 5.2, the most important thermal parameters of the materials are displayed.

Si linear term of thermal conductivity	295 W/(mK)
Si quadratic term of thermal conductivity	-0.491 W/(mK^2)
SiO_2 thermal conductivity	1.38 W/(mK)
Si specific heat	1.628 x 10^6 J/m^3K
SiO_2 specific heat	4.180 x 10^6 J/m^3K
Isolating Air thermal conductivity	2.4 x 10^-3 W/(mK)
Isolating Air specific heat	1 x 10^4 J/m^3K
Water specific heat	4.184 x 10^6 J/m^3K
Water thermal conductivity	0.58 W/(mK)

Table 5.2: Material thermal properties.

The recursive solving method is stopped when the normalized error given by Equation 5.1, is smaller than 10^{-6}.

$$Error_{norm} = \frac{norm(T(i) - T(i-1))}{norm(T(i))} \tag{5.1}$$

After solving the equation set, the temperature of each cell, in which the 3D stack was split, is returned in the resultant thermal profile of the optimized 3D stack as depicted in Figure 4.28. This file contains not only the temperatures but also the floorplan characteristics, in order to describe the functional units distribution to the μchannel deployment optimizer.

5.4. Homogeneous Floorplan.

Homogeneous architectures have been traditionally used as a way to parallelize computation works in multi-core systems. In these environments the microarchitecture of the processing units is invariant making the programming of the software easier, because the programmer deals only with one processing architecture. The main feature of an homogeneous scheme is that it allows a higher flexibility and makes the system scalability trivial, only the same kind of processing units should be added to the design. Besides it has been proven by the experience of multi-core systems, such as [132] or [141], that homogeneous computation is more suited for general-purpose tasks.

The previous 3D design guidelines offered some rules to conduct the thermal aware placement, however, in MPSoCs, when the number of functional units increases, it is not easy to apply these "common sense" rules in order to reduce the thermal activity, because the impact of one functional unit to others is more sensitive due to the reduction of the free space.

In this section it will be presented the results obtained by our floorplanner with an homogeneous scenario of up to 128 cores. The study will be made using an homogeneous floorplan in which every core is assumed to have the same power density (it must be reminded that worst case is contemplated).

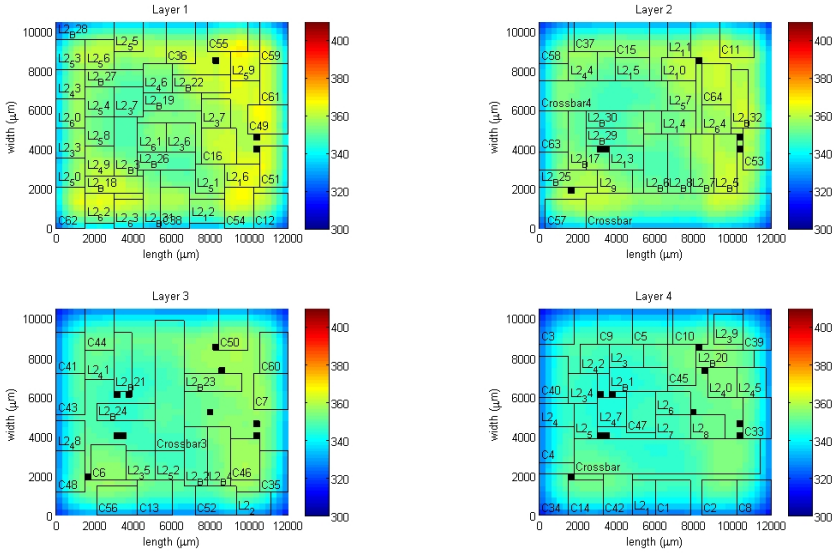


Figure 5.7: Thermal behavior output.

Homogeneous designs simplify the optimization placement because the size and the behavior of the functional units considered as heat sources is exactly the same. With this principle the floorplanner will follow the same strategy with each of the cores of the design.

5.4.1. Thermal Profile.

The simulator has been developed with a graphical thermogram viewer as the one presented in Figure 5.7. These Figures will show the temperatures of the active layers, those in which the functional units are placed, using a color map schema which is given in the legend. The representation is made filling each of the cells with the color that corresponds with its temperature. In order to make it easier to understand the thermal improvement of the optimizations, the scale for each design is kept constant.

The functional units are represented as blocks with its name that comes from the original Niagara design. If there were TSVs deployed they are represented as black spots. If there were any isolation air channels these would be represented with '*'.

5.4.2. Results for the Original.

In this first thermal profiling it will be shown the thermal behavior of several designs with an increasing number of cores in the stack. The devices that will be studied are built with 16, 48, 64 and 128 cores distributed in 2, 4, 5 and 9 tiers respectively. These designs, as was previously presented, will be based on Niagara 2 and Niagara 3 floorplans as the ones depicted in

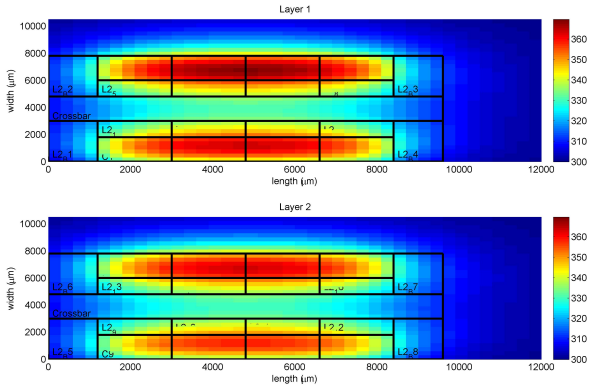


Figure 5.8: Thermal results for the original 16-core system.

Figure 5.5. Combining these floorplans in different tiers, different multi core system can be created.

The results obtained by our simulator can be seen in Figures 5.8, 5.9, 5.10 and 5.11. These thermal profiles have been calculated in order to exhibit how the thermal problem is risen when the number of functional units is increased, which entails an increment in the power dissipated in the chip.

From the first visual analysis of the figures, it can be noticed that there are high temperature peaks in certain regions of the area where hot spots are created. This problem is exacerbated in firsts layers, as was seen in the 3D design guidelines given in 5.1, where the stack is not able to dissipate heat to the environment.

The thermal problems shown in these configurations, affects the performance and the reliability of the chip. The affections that were exhibited in Chapter 2 such as electromigration due to thermal gradients and hot spots could be found in these distributions due to their high temperature. As the system scales up, the thermal problem is obviously higher since the power dissipated in the stack is increased. Besides there are more inner layers and the chip is not able to cool down the temperature.

The thermal behavior exhibited by these four configurations, reflects the need of an automatic design tool to cope with 3D thermal issues. In the following it will be shown how the optimizations carried out by our floorplanner, are able to mitigate the thermal distribution of the 3D stacks.

5.4.3. Optimizing the Placement of Functional Units.

Since the placement of the functional units, as shown before, is of vital importance in the thermal behavior of the 3D stack, it has to be done carefully. Due to the big number of functional units to be placed, the process has to be addressed automatically.

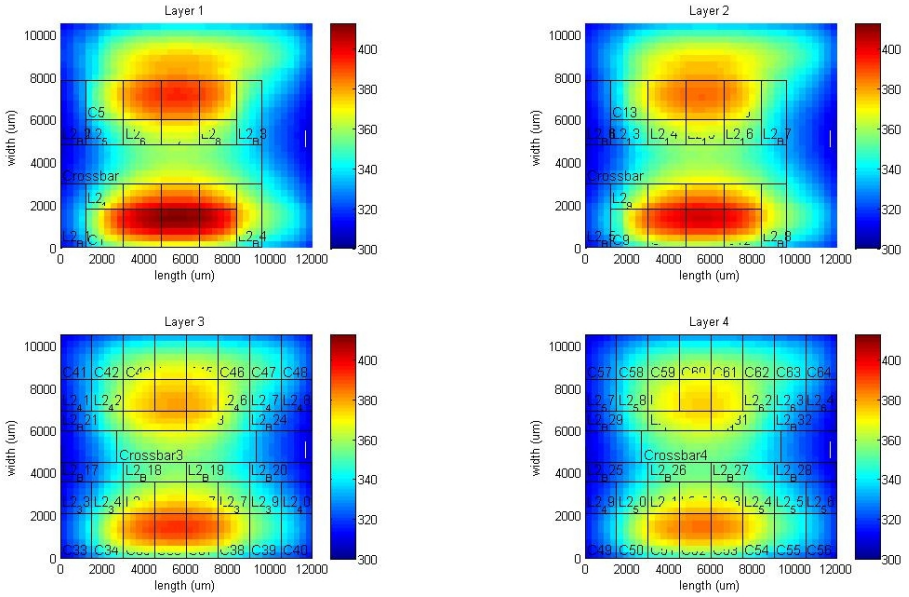


Figure 5.9: Thermal results for the original 48-core system.

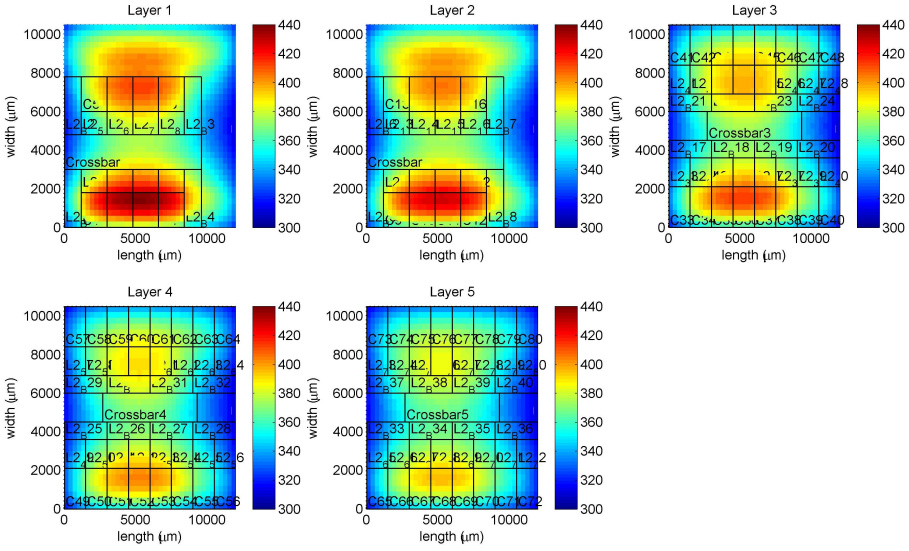


Figure 5.10: Thermal results for the original 64-core system.

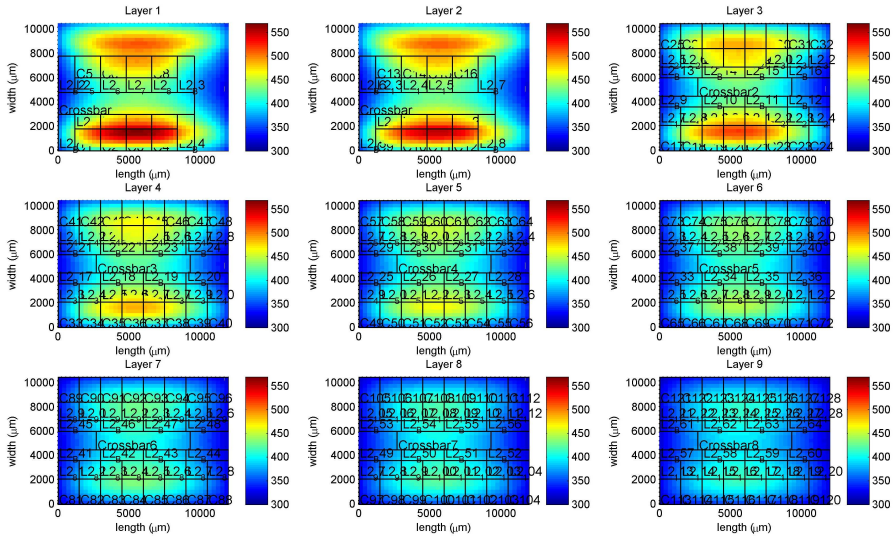


Figure 5.11: Thermal results for the original 128-core system.

If the original designs that were used before are optimized using our floorplanner the thermal metrics improve considerably. The thermal results obtained by our floorplanner are depicted in Figures 5.12, 5.13, 5.14 and 5.15, for the same cases of 16, 48, 64 and 128 cores respectively.

The previously defined thermal metrics, the mean temperature, thermal gradient and maximum temperature for every layer of the configuration, have been calculated. The comparison with the baseline homogeneous systems show that the floorplanner is capable of optimizing every studied thermal value.

This can be explained because our floorplanner spaces heat sources (cores) as much as possible, trying to place them at the border of the chip, helping on the cooling down of the cores. The floorplanner also takes into account vertical heat spread, and each layer will have a different layout, avoiding placing heat sources one over the other. As can be seen in the optimized floorplans the cores are mainly placed in the first and last layer, leaving inner layers with heat sinks. Using this approach heat is spread equally in all the chip achieving big reductions in the maximum temperature and hence thermal gradients. However the improvement in the mean temperature is not as perceptible as the other thermal metrics. At this point it has to be reminded that the optimizer seeks the optimization of maximum temperature and wire length without impacting on the mean temperature negatively.

Tables 5.3, 5.4 and 5.5, exhibit the comparison of the the thermal metrics, such as mean temperature, maximum temperature an thermal gradient for each active layer between the original and the optimized designs.

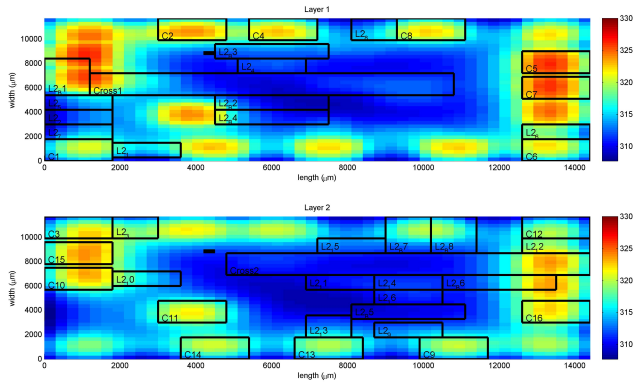


Figure 5.12: Thermal results for the optimized 16-core system.

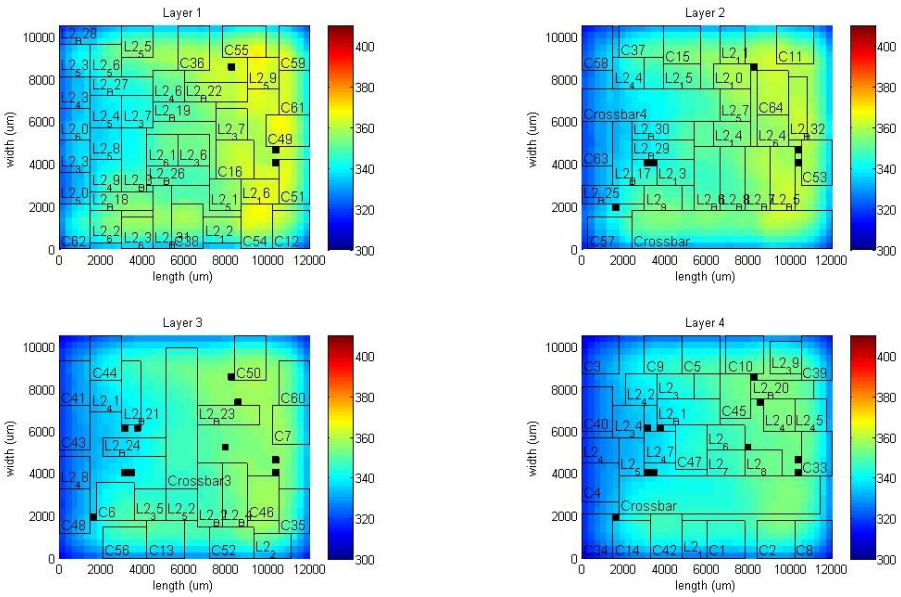


Figure 5.13: Thermal results for the optimized 48-core system.

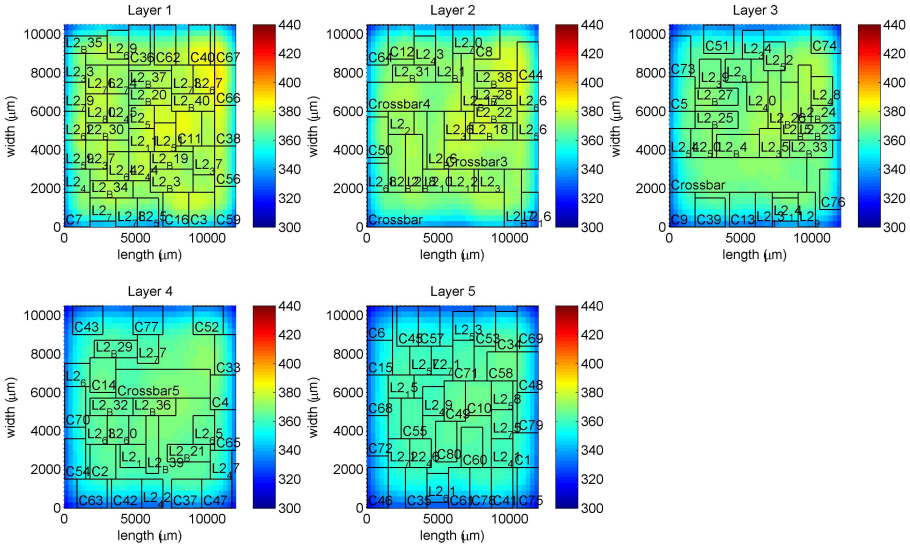


Figure 5.14: Thermal results for the optimized 64-core system.

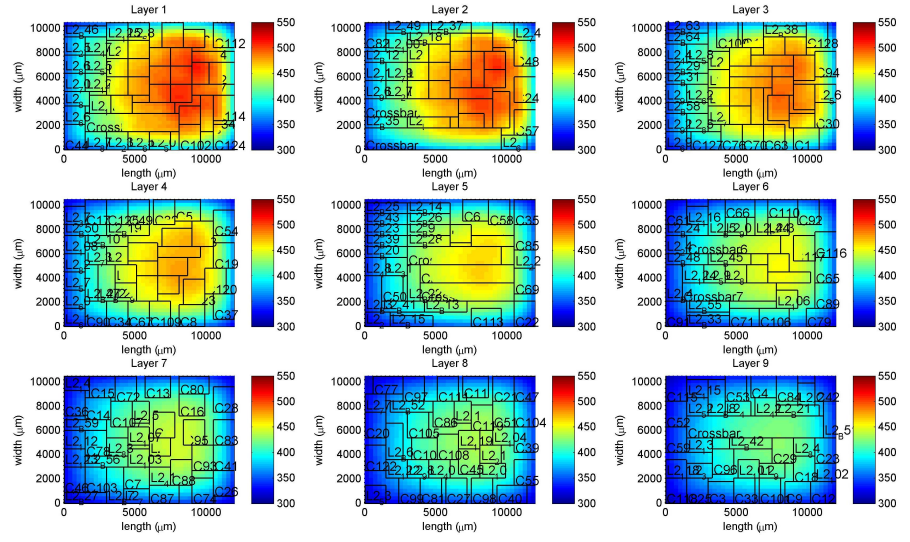


Figure 5.15: Thermal results for the optimized 128-core system.

Design	*16 cores*		*48 cores*		*64 cores*		*128 cores*	
	Orig.	Opt.	Orig.	Opt.	Orig.	Opt.	Orig.	Opt.
Layer 1	368.7	326.4	413.0	369.6	438.1	390.8	566.3	511.7
Layer 2	363.3	324.0	403.8	363.8	427.4	384.5	550.7	508.0
Layer 3	N/A	N/A	393.7	358.7	413.7	376.6	519.0	493.1
Layer 4	N/A	N/A	386.2	355.1	403.5	371.4	487.9	478.1
Layer 5	N/A	N/A	N/A	N/A	395.6	367.0	464.6	465.3
Layer 6	N/A	N/A	N/A	N/A	N/A	N/A	447.6	454.0
Layer 7	N/A	N/A	N/A	N/A	N/A	N/A	434.1	443.7
Layer 8	N/A	N/A	N/A	N/A	N/A	N/A	423.0	434.3
Layer 9	N/A	N/A	N/A	N/A	N/A	N/A	414.3	425.5

Table 5.3: Maximum temperature comparison.

Design	*16 cores*		*48 cores*		*64 cores*		*128 cores*	
	Orig.	Opt.	Orig.	Opt.	Orig.	Opt.	Orig.	Opt.
Layer 1	68.3	17.9	103.1	47.8	124.7	65.7	241.4	182.3
Layer 2	62.8	16.2	94.0	44.6	114.2	61.9	226.1	179.7
Layer 3	N/A	N/A	84.2	42.4	100.8	57.4	194.8	168.2
Layer 4	N/A	N/A	77.1	40.4	91.0	54.6	164.3	157.6
Layer 5	N/A	N/A	N/A	N/A	83.6	51.7	141.6	147.9
Layer 6	N/A	N/A	N/A	N/A	N/A	N/A	126.5	139.0
Layer 7	N/A	N/A	N/A	N/A	N/A	N/A	115.9	130.4
Layer 8	N/A	N/A	N/A	N/A	N/A	N/A	106.9	122.2
Layer 9	N/A	N/A	N/A	N/A	N/A	N/A	99.4	114.4

Table 5.4: Thermal gradient comparison.

From the results reflected in the tables it can be derived that the optimizer decreases the maximum temperature in all the active layers for all the designs but the 128 core system for the upper layers. This is because the replacement of functional units is made to ensure a more homogeneous thermal distribution coping with the high temperatures reached in the firsts layers. As can be seen, for the 128 core system, the maximum temperature of the first layer is reduced in 54 degrees. The thermal gradient follows the same tendency, while the mean temperature is almost maintained in the optimized designs.

Another statistical parameter that can be studied from the thermal behavior of the designs is the thermal deviation. As shown in Table 5.6, the deviation of temperatures is clearly reduced. This reduction determines a more homogeneous thermal distribution, which is translated into a reduced reliability risk and diminished leakage currents.

Design	*16 cores*		*48 cores*		*64 cores*		*128 cores*	
	Orig.	Opt.	Orig.	Opt.	Orig.	Opt.	Orig.	Opt.
Layer 1	324.7	315.7	356.8	353.3	371.3	368.8	420.7	420.1
Layer 2	323.2	314.6	353.9	350.2	368.0	365.6	418.0	417.2
Layer 3	N/A	N/A	349.9	346.5	362.5	360.3	408.7	408.5
Layer 4	N/A	N/A	346.7	343.4	358.0	355.9	398.1	397.5
Layer 5	N/A	N/A	N/A	N/A	354.2	352.2	389.3	388.2
Layer 6	N/A	N/A	N/A	N/A	N/A	N/A	381.8	380.2
Layer 7	N/A	N/A	N/A	N/A	N/A	N/A	375.5	373.3
Layer 8	N/A	N/A	N/A	N/A	N/A	N/A	370.1	367.3
Layer 9	N/A	N/A	N/A	N/A	N/A	N/A	365.5	362.2

Table 5.5: Mean temperature comparison.

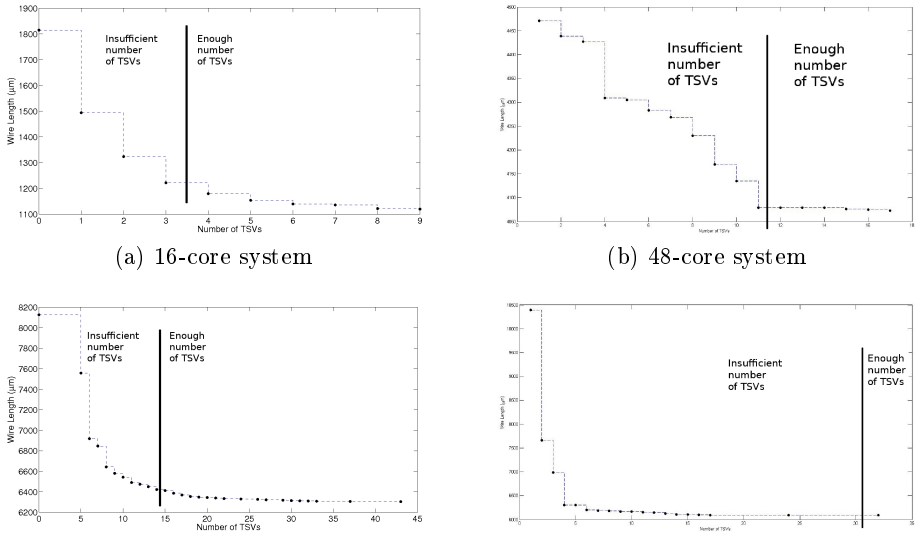
Design	*16 cores*	*48 cores*	*64 cores*	*128 cores*
Original	18.7	21.3	24.4	41.2
Optimized	4.1	8.8	12.7	39.1

Table 5.6: Thermal deviation (K)

5.4.3.1. Wire results.

The optimization of the placement of the TSVs is carried out using a multi-objective genetic algorithm during the placement of functional units, as explained in Section 4.3.2.

The algorithm gives the designer a Pareto front approximation with the number of TSVs and chip wire length. As was said, it will be the designer who will choose which solution is more convenient in every case, depending on the economic cost and technological issues, and the design benefits obtained with the inclusion of more TSVs.



(a) 16-core system

(b) 48-core system

(c) 64-core system

(d) 128-core system

Figure 5.16: Pareto front approximation for 16, 48, 64 and 128-core systems.

Figures from 5.16a to 5.16d show the Pareto front approximation for the optimized homogeneous case. As can be seen in the figures, the tendency of wire length with the number of TSVs follows an almost negative exponential tendency.

Every thermal improvement entails an overhead in the performance of the IC because of communication delays caused by the increase in wire length. On the other hand, the big savings reached in the maximum temperature (up to 54 degrees) justify the overhead in wiring.

Table 5.7 shows the wire length associated with the 3D system when the minimum number of needed TSVs is chosen. As can be seen, the overhead incurred by the floorplanner in the worst case scenario (128 cores) has been a 63 % when compared to the original homogeneous distribution and 51 % for the heterogeneous system. It must be noted, that the original design is optimized for for the 2D scenario and that is why such an overhead is obtained. However our designs achieve great results and constitute feasible designs in terms of wiring if they are compared with the 2D scenario writen in the table as *2D original scenario*.

This optimization process takes some days of computational cost. Since the optimization is carried out during the design phase the duration of the optimization flow is not really relevant. The time invested in the execution of the optimization flow is negligible compared to the benefits that it entails. However, the work initiated by [6], [5], based on the one proposed by this

Scenario	*16 cores*	*48 cores*	*64 cores*	*128 cores*
2D original scenario	449	1448	2214	5984
Original	329	1319	1636	3187
Optimized	399	1459	2012	4987

Table 5.7: Wire Length (mm)

thesis, have direct their research to find parallelization methods to make the algorithm faster.

5.4.4. Optimization of μchannels.

The high temperatures reached in the baseline stack can be managed including liquid micro channels, as an effective way to decrease temperature in inner layers. In the previously presented scenarios, a different number of channels have been deployed in order to show the thermal benefits of including active cooling systems.

The considered ratio for the cooling mechanisms is 0.25 μchannels/core, which entails 4, 12, 16 and 32 channels for the previously presented optimized designs of 16, 48, 64 and 128 cores respectively.

In order to show the importance of μchannel optimization two deployment strategies will be taken, homogeneous placement of μchannels and an optimized location search. Figures 5.17, 5.18, 5.19 and 5.20, show the thermal maps after having optimized the placement of the microchannels in the already optimized designs. Liquid μchannels are deployed in those positions marked with an arrow and flow from the bottom to the top of the layer. These channels are placed by the optimizer in those regions where the temperature reached high values. As can be seen, with this optimized distribution of μchannels, the new thermal floorplans exhibit a more homogeneous thermal distribution because the channels not only cool down the system but also contribute to spread heat from hot areas to those regions where the temperature is lower.

In Table 5.8 the thermal metrics of the possible optimizations are compared for every design. The first value $Before$ corresponds to the optimized scenario before the deployment of any liquid channel. $AfterHom$ represents the values for the homogeneous placement of liquid channels while $AfterOpt$ gives the thermal behavior for the optimized placement of liquid channels.

Finding optimal distribution for liquid channels improves not only the thermal metrics but also fabrication costs, because wider channels can be built by replacing two or more thinner channels and also reducing pumping energy.

	16 cores	*48 cores*	*64 cores*	*128 cores*
$Max._{Before}$	326.4	369.6	390.8	511.7
$Max._{AfterHom}$	326.2	362.3	377.7	446.4
$Max._{AfterOpt}$	324.6	355.6	371.1	406.2
$Mean_{Before}$	315.2	348.4	360.6	390.5
$Mean_{AfterHom}$	313.8	335.7	345.5	358.3
$Mean_{AfterOpt}$	313.3	333.8	340.5	349.3
$Grad._{Before}$	17.9	47.8	65.7	182.0
$Grad._{AfterHom}$	17.8	44.3	57.5	123.7
$Grad._{AfterOpt}$	17.7	38.9	50.0	77.9

Table 5.8: Thermal metrics before and after μchannel deployment.

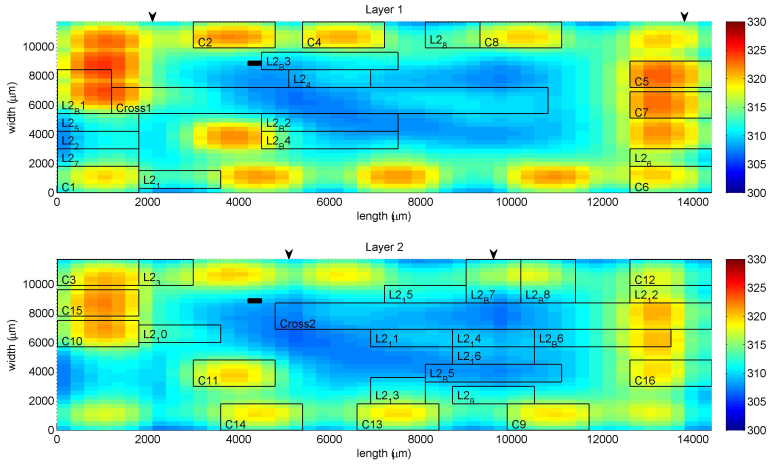


Figure 5.17: Thermal results for the optimized 16-core system with 4 μchannels.

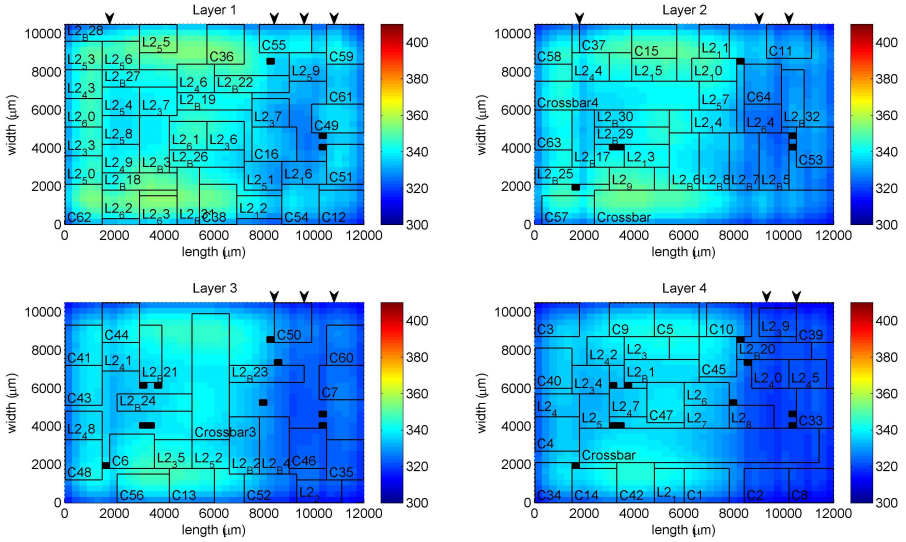


Figure 5.18: Thermal results for the optimized 48-core system with 12 μchannels.

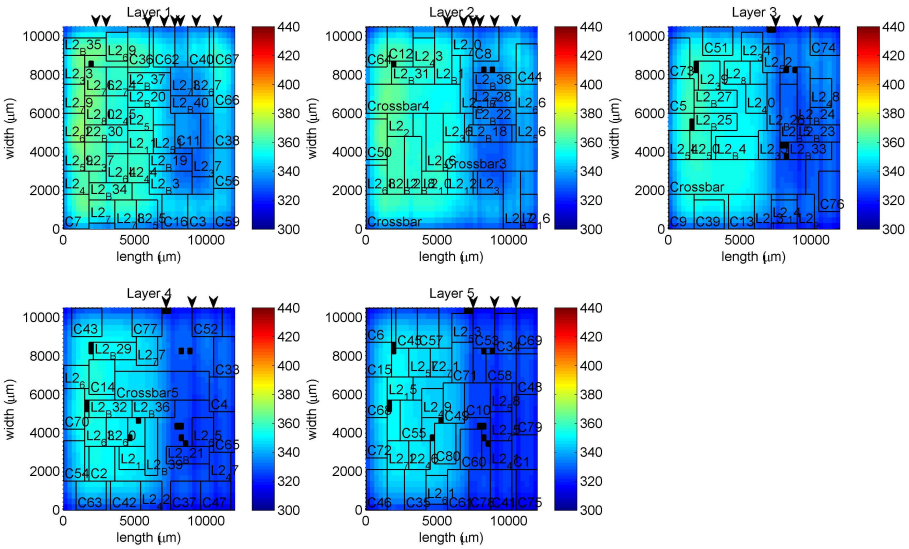


Figure 5.19: Thermal results for the optimized 64-core system with 16 μchannels.

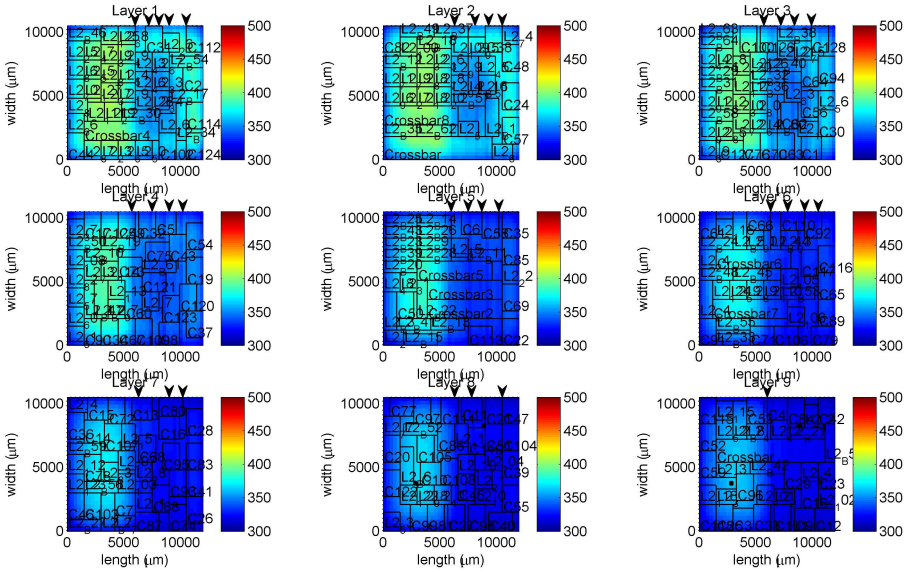


Figure 5.20: Thermal results for the optimized 128-core system with 32 μchannels.

Liquid channel optimized placement achieves very good results, but these thermal results can be enhanced adding several optimizations such as functional units placement.

In the following the liquid channel optimizer will be applied to our optimized floorplan. As was previously commented. Our liquid placement optimizer evaluates not only hot areas but also TSVs location, because liquid channels cannot go through TSVs. With the placement of liquid μchannels, it can be seen that there are no longer hot areas. Optimizing both, placement of functional units, and the placement of liquid microchannels. In the best case a reduction of 57K in maximum temperature, 33K in mean temperature and 66K in the gradient has been achieved when compared with the original scenario.

5.4.4.1. Execution time.

The μchannel optimization process is much easier than the optimization of the placement of functional units, thus, the execution times are much lower, reducing these optimization costs from days to minutes. In Table 5.9 the optimization times are provided.

16 cores	*48 cores*	*64 cores*	*128 cores*
90	113	133	224

Table 5.9: Execution Time (s)

5.5. Heterogeneous Floorplan.

Heterogeneous computing systems refer to electronic systems that use a variety of different types of computational units. A computational unit could be a general-purpose processor (GPP), a special-purpose processor (i.e. digital signal processor (DSP) or graphics processing unit (GPU)), a co-processor, or custom acceleration logic (application-specific integrated circuit (ASIC) or field-programmable gate array (FPGA)). In general, a heterogeneous computing platform consists of processors with different instruction set architectures (ISAs).

Heterogeneous systems have got a lot of importance due to the need for high specific performance and highly reactive systems that interact with other environments (audio/video systems, control systems, networked applications, etc).

Frequency scaling has proven to be an effective way to increase performance without requiring structural changes or custom hardware acceleration, however the increased need of higher performances for certain applications makes this technique pretty obsolete.

Another method of gaining extra performance out of computing systems is to introduce additional specialized resources, thus making a computing system heterogeneous. This technique allows to include multiple types of processing elements in the same package. Each of these elements could be dedicated to perform the tasks which it is best suited for.

The addition of extra, independent computing resources necessarily allows most heterogeneous systems to be considered parallel computing, or multi-core (computing) systems.

The level of heterogeneity in modern computing systems gradually rises as increases in chip area and further scaling of fabrication technologies allow for formerly discrete components to become integrated parts of a system-on-chip, or SoC. For example, many new processors include built-in logic for interfacing with other devices (SATA, PCI, Ethernet, RFID, Radios, UARTs, and memory controllers), as well as programmable functional units and hardware accelerators (GPUs, cryptography co-processors, programmable network processors, A/V encoders/decoders, etc.).

5.5.1. Results for the Original Heterogeneous Design

As was previously done with the homogeneous case, the heterogeneous design is also based in the Niagara 2 and Niagara 3. However some processing units have been swapped by others with a different technology and, more important for the thermal purposes, different power consumption which entails different design patterns.

This second scenario consists on an heterogeneous system where the same number of cores are deployed. The heterogeneous system is composed by SPARC and Power6 cores, with a ratio of 3/1. This setup will show the optimized thermal profile that can be expected when multiple core architectures are considered, as well as the extra optimization opportunities that the floorplanner will find. The study could be much wider, however this PhD Thesis wants to show the different possibilities offered in 3D design and their particularities.

In this scenario the 16 core system will be obviated, and only simulations and optimizations for the 48, 64 and 128 cases will be conducted. The original heterogeneous scenario is built swapping some of the cores of the initial Niagara distribution in order to emulate an heterogeneous core distribution.

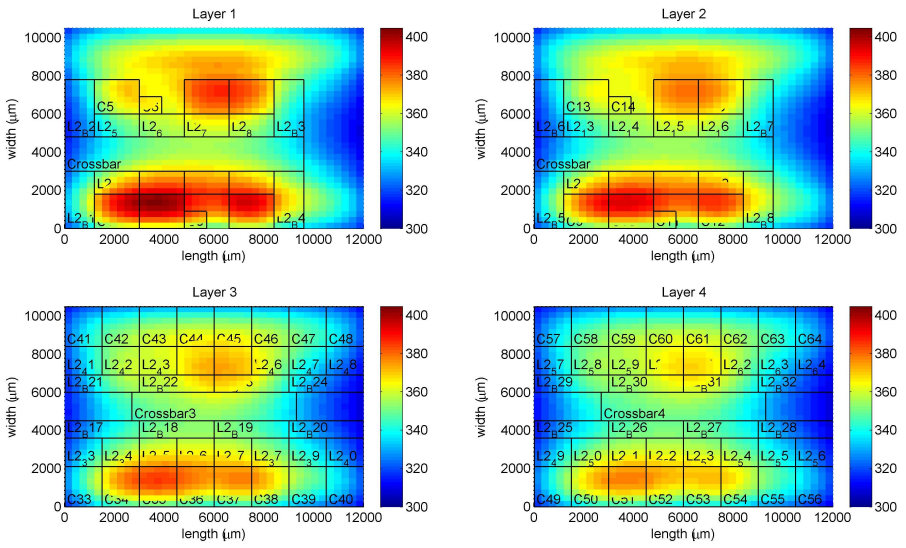


Figure 5.21: Thermal results for the original heterogeneous 48-core system.

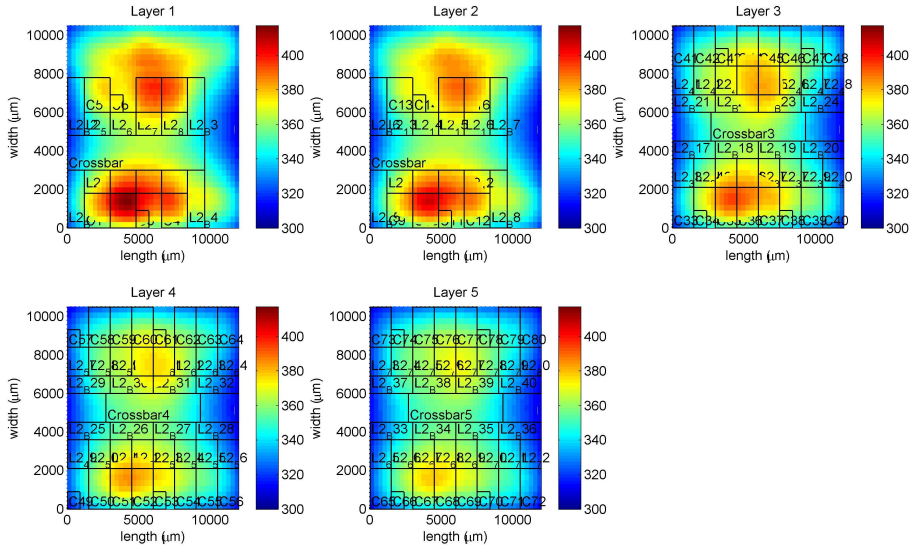


Figure 5.22: Thermal results for the original heterogeneous 64-core system.

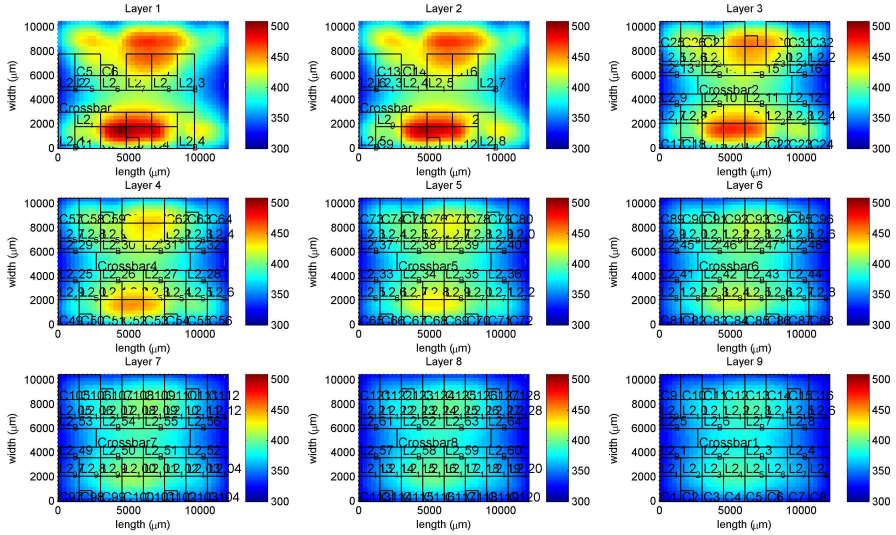


Figure 5.23: Thermal results for the original heterogeneous 128-core system.

Although the power density of the new POWER 6 cores is lower than the previous SPARC there still exist several hot spots, over all in the first layers as can be seen in Figures 5.21, 5.22 and5.23. However the lower power dissipation in these heterogeneous floorplans is translated into a decrease in the thermal metrics.

5.5.2. Optimization of the Placement

Again, the obtained thermal results of the original heterogeneous designs, invite to test our proposed optimization algorithms in order to improve the thermal behavior of the floorplan. In the following, as was previously made with the homogeneous cases, the tabulated results and figures of the optimized designs will be shown. When this set of data is compared with the homogeneous case it can be derived that due to the lower power dissipation, the algorithm is able to find better locations for the power sources (processors), increasing the temperature differences between the original and the optimized scenarios even more than in the homogeneous case.

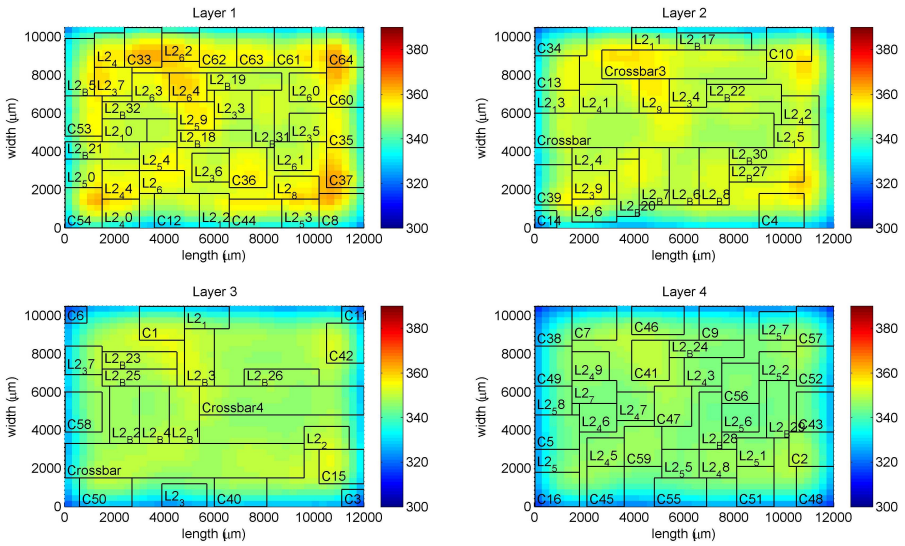


Figure 5.24: Thermal results for the heterogeneous optimized 48-core system.

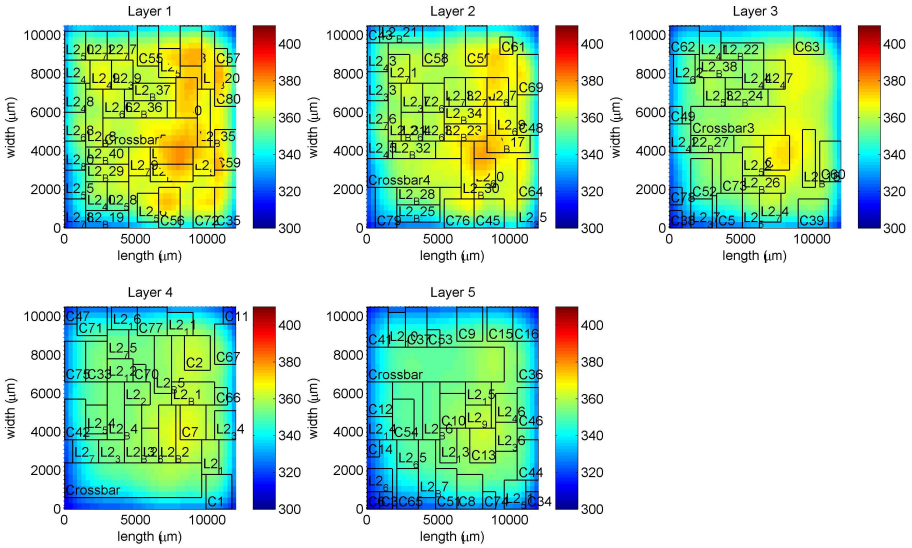


Figure 5.25: Thermal results for the heterogeneous optimized 64-core system.

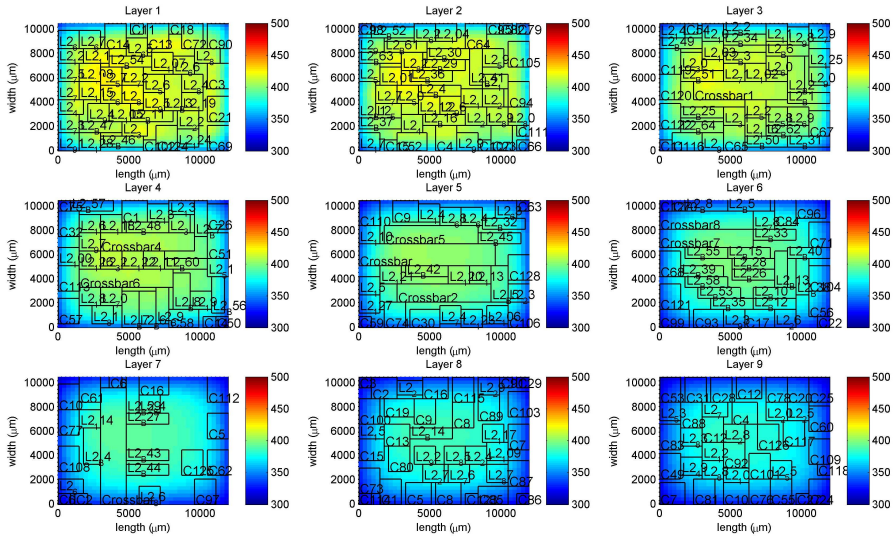


Figure 5.26: Thermal results for the heterogeneous optimized 128-core system.

Figures 5.24, 5.25 and 5.26 show the thermograms for the optimized designs. As happened with the homogeneous designs, temperature is distributed homogeneously throughout the chip alleviating thermal stresses.

Tables 5.10, 5.11 and 5.12, show the original versus the optimized results

Design	*48 cores*		*64 cores*		*128 cores*	
	Orig.	Opt.	Orig.	Opt.	Orig.	Opt.
Layer 1	385.4	357.5	416.0	380.6	510.8	426.1
Layer 2	377.5	353.3	406.4	378.1	500.6	424.8
Layer 3	370.4	349.5	394.0	371.7	476.6	420.8
Layer 4	365.2	364.4	385.2	366.7	452.4	412.6
Layer 5	N/A	N/A	378.7	362.4	434.0	405.2
Layer 6	N/A	N/A	N/A	N/A	420.7	398.8
Layer 7	N/A	N/A	N/A	N/A	409.8	393.0
Layer 8	N/A	N/A	N/A	N/A	401.1	387.6
Layer 9	N/A	N/A	N/A	N/A	394.2	382.4

Table 5.10: Maximum temperature comparison.

Design	*48 cores*		*64 cores*		*128 cores*	
	Orig.	Opt.	Orig.	Opt.	Orig.	Opt.
Layer 1	76.9	40.4	103.9	63.9	188.6	90.4
Layer 2	69.2	38.0	94.4	63.0	178.6	90.0
Layer 3	62.2	36.6	82.3	59.3	155.1	91.1
Layer 4	57.4	34.8	73.9	55.9	131.4	88.9
Layer 5	N/A	N/A	67.8	52.7	113.6	85.7
Layer 6	N/A	N/A	N/A	N/A	103.3	82.2
Layer 7	N/A	N/A	N/A	N/A	94.9	78.5
Layer 8	N/A	N/A	N/A	N/A	88.0	74.6
Layer 9	N/A	N/A	N/A	N/A	82.2	70.4

Table 5.11: Thermal gradient comparison.

for maximum temperature, thermal gradient and mean temperature.

From these data, it can be seen how our floorplanner, again, is able to reduce the maximum temperature of the heterogeneous design in 84 degrees in the best case (the first layer of the 128-core floorplan=. These results outperforms the ones achieved in the homogeneous case and this is because, our floorplanner is able to manage much more options since there are heat sources with a different power dissipation rate.

Our floorplanner in the heterogeneous case, is not only able to outperform the thermal metrics of the homogeneous designs but also achieve a better distribution of the temperature throughout the 3D stack as can be derived from Table 5.13, where the deviation data are shown.

Wire optimization and execution time are very similar in both scenarios, because the heterogeneous designs only change the power dissipation and

Design	*48 cores*		*64 cores*		*128 cores*	
	Orig.	Opt.	Orig.	Opt.	Orig.	Opt.
Layer 1	345.1	342.8	358.8	358.3	403.0	401.2
Layer 2	342.8	340.4	356.2	355.4	400.6	398.7
Layer 3	339.7	337.4	351.8	351.1	393.1	392.5
Layer 4	337.2	335.1	348.1	347.4	384.4	384.1
Layer 5	N/A	N/A	345.0	344.4	377.0	376.8
Layer 6	N/A	N/A	N/A	N/A	370.7	370.6
Layer 7	N/A	N/A	N/A	N/A	365.3	365.3
Layer 8	N/A	N/A	N/A	N/A	360.7	360.8
Layer 9	N/A	N/A	N/A	N/A	356.8	356.9

Table 5.12: Mean temperature comparison.

Design	*48 cores*	*64 cores*	*128 cores*
Original	16.5	19.0	29.5
Optimized	8.3	12.4	21.5

Table 5.13: Thermal deviation (K)

the size of some cores in the floorplan, maintaining the number of functional units. The number of functional units impacts directly on the algorithm execution time while connections among functional units affect the wire length. Since these variables are kept constant, the differences between the homogeneous and the heterogeneous scenario in the execution time and wire length are negligible.

5.5.3. Optimization of the μchannels

As can be seen, the obtained results are too far from being considered as acceptable. Once that the importance of optimizing the distribution of μchannels has been explained, the thermal results for an optimized deployment of 12, 16 and 32 channels for the previously presented optimized designs of 48, 64 and 128 cores respectively will be presented.

Figures 5.27, 5.28 and 5.29, depict the thermal results when the optimized number of liquid channels has been placed. As can be seen in these figures, the new thermal profile changes significantly and those areas in which the cores are more concentrated become more susceptible to have a liquid channel which minimizes the impact of power dissipation in that area.

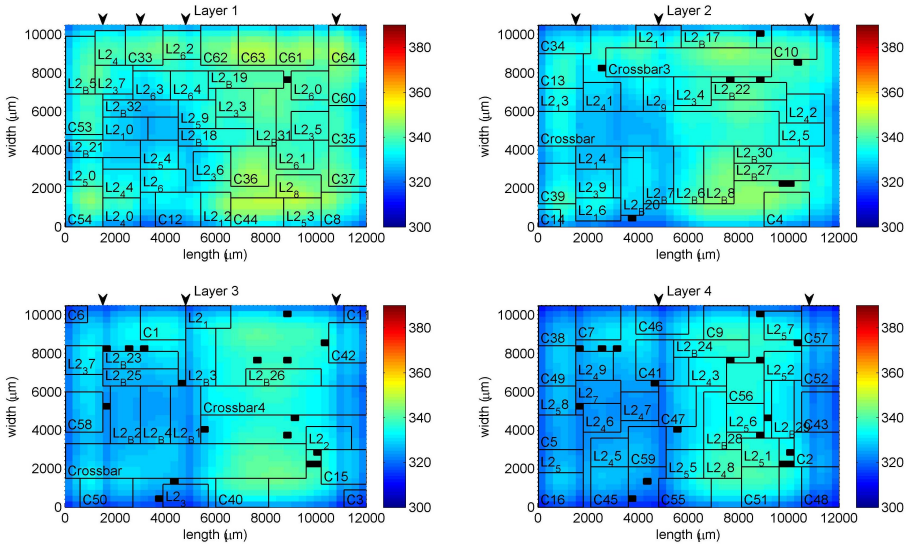


Figure 5.27: Thermal results for the heterogeneous optimized 48-core system with 12 μchannels.

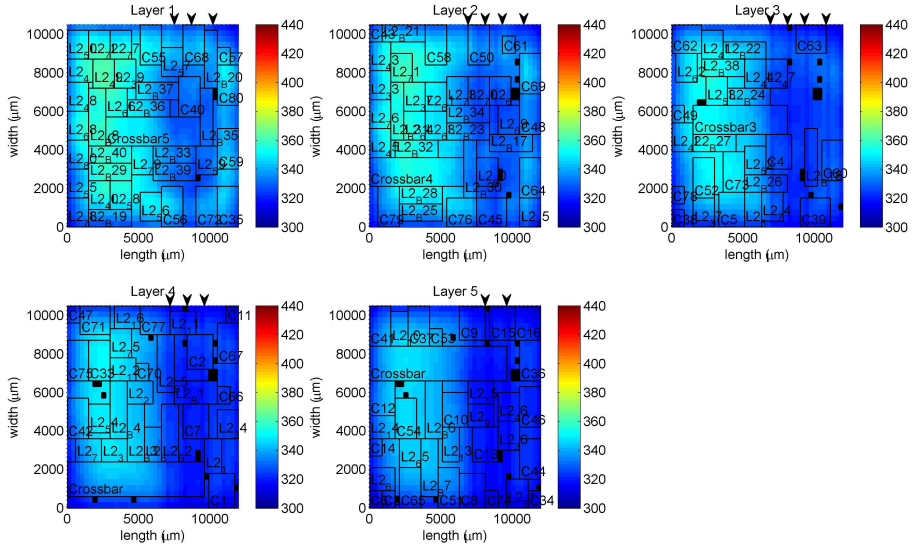


Figure 5.28: Thermal results for the heterogeneous optimized 64-core system with 16 μchannels.

	48 cores	*64 cores*	*128 cores*
$Max._{Before}$	357.5	380.6	426.1
$Max._{After}$	349.2	362.6	388.9
$Mean._{Before}$	338.9	351.3	378.5
$Mean._{After}$	331.4	334.4	338.6
$Grad._{Before}$	40.4	63.9	90.4
$Grad._{After}$	30.6	48.3	60.7

Table 5.14: Thermal metrics before and after μchannel deployment.

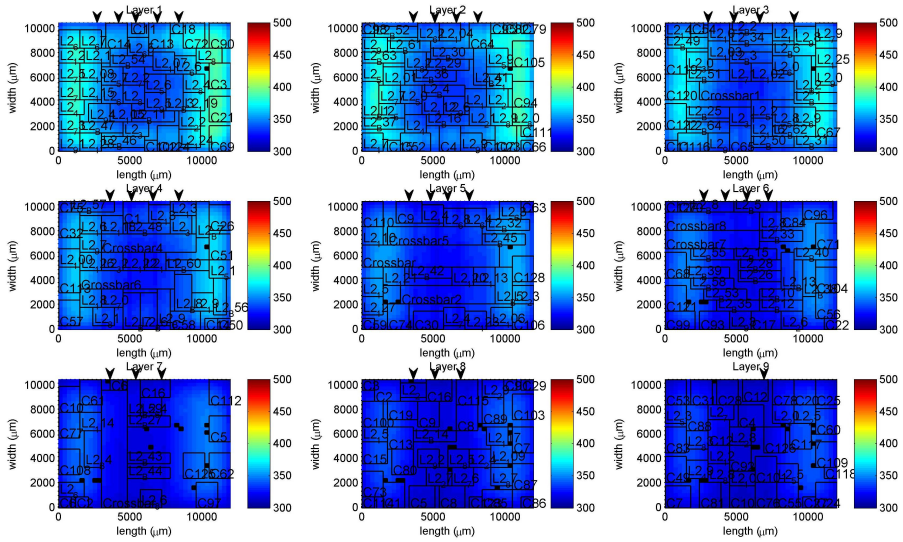


Figure 5.29: Thermal results for the heterogeneous optimized 128-core system with 32 μchannels.

With the placement of liquid microchannels the reached temperatures are acceptable for an electronic device. As can be seen from the results in Table 5.14, the deployment of the μchannels alleviates the thermal problem, reducing the maximum temperature in 38 additional degrees when comparing with the already optimized design. But not only the maximum temperature and the thermal gradient is affected by the active cooling system, the mean temperature of the 3D stack is also reduced considerably.

As happened with the optimization of the placement of functional units, the execution time of the μchannel algorithm depends only on the number of channels and the complexity of the floorplan. Since these variables are kept constant in both, homogeneous and heterogeneous scenarios, the execution

Representation	*Algorithm*	*Max. Temp.(°C)*	*Wirelength(mm)*	*Obj.*
Original	-	**140**	**1319**	-
CBA [29]	SA	129.6	1251	0.93
SP [118]	SA	126.3	1296	0.92
GPE [11]	GA	134.5	1282	0.96
Proposal- MFA	***GA***	***96.6***	***1459***	0.89

Table 5.15: Thermal comparison of different algorithms for the placement of functional units.

time difference is negligible.

5.6. Comparison with other State of the Art Optimization Algorithms

In Section 5.4 and Section 5.5 we have presented the results obtained for two different scenarios, based on certain 3D thermal-aware guidelines and assuming that our floorplanner offered better results than other strategies that can be found in the literature. In this section a comparison between our floorplanner and some of the most important thermal aware strategies that were previously presented in 4.3.1.2 will be shown.

For this comparison the same thermal metrics will be taken into account. The wirelength of the final floorplans is also presented as an important part of the thermal analysis.

The baseline scenario for the comparison will be the 48 core design previously described in Section 5.4.2, and its thermal behavior is depicted in Figure 5.9.

Based on this original scenario several optimizations using different algorithms and representations were made. In Table 5.15, the results obtained by other implemented algorithms is shown. These methods are extracted from the literature. They have been evaluated using a weighted sum of maximum temperature and wirelength. The expression that the algorithms evaluate is given by 5.2, which is a weighted sum of both objectives, maximum temperature and wirelength, taking as a reference the results obtained with the original not optimized design.

$$Obj. = 0.5\frac{W}{W_{original}} + 0.5\frac{Tmax(^oC)}{Tmax_{original}(^oC)} \tag{5.2}$$

As can be seen our proposal outperforms the thermal results obtained by other algorithms that use other representations or optimization processes

improving the combined objective given by the maximum temperature and wire length.

5.7. Proposed Thermal-Aware Architecture with Air Channels

The previously presented results show how our thermal-aware floorplanning algorithms are able to reduce the temperature of several realistic systems that have been modified to contemplate the 3D case. These algorithms work evaluating the optimal positions of the functional units and then, the optimal positions of the liquid μchannels.

In this section, the objective consists of showing, how, our proposed technique of creating thermal regions, can result in a more profitable and cheaper designs when it is combined with the previous presented optimization techniques.

The creation of thermal regions is achieved by the inclusion of air isolation channels in the active layers which make the heat concentrate in certain areas of the stack. Those areas will be then categorized in hot and warm regions. Our algorithm will place those functional units which dissipate more power density in the hot regions and the heat sinks in the warm areas. This will create a two temperature design, but it will force the liquid channel placement to the warm regions alleviating the thermal problem with less μchannels as will be shown later.

In order to show how the creation of temperature regions works in the whole optimization process the 48 core design will be taken. In figure 5.30 initial distribution without liquid channels is presented in order to remind how hot spots appear due to the disposition of the functional units in the stack.

This initial design has been optimized using our floorplanner, with the initial configuration of one air channel per layer. This configuration has been taken based on the experience and free space in the 3D stack. One air channel per layer is enough to take advantage of the thermal regions. The thermal behavior of this optimized scenario is depicted in Figure 5.31b, where $*$ symbols represent the existence of an air channel. As can be seen, the floorplanner has located functional units with a high power consumption in hot regions (left side for layers 1 and 3; right side for layer 2 and 4) and functional units with lower power densities in warm regions.

As may be obvious this new optimized scenario presents worse thermal metrics than the previous optimized design without air channels, because heat sources are placed closer, creating hotter areas not only in their active layer but also in upper and lower ones, because of vertical heat transfer. The stack is not able to dissipate heat to the environment as easy as the previous

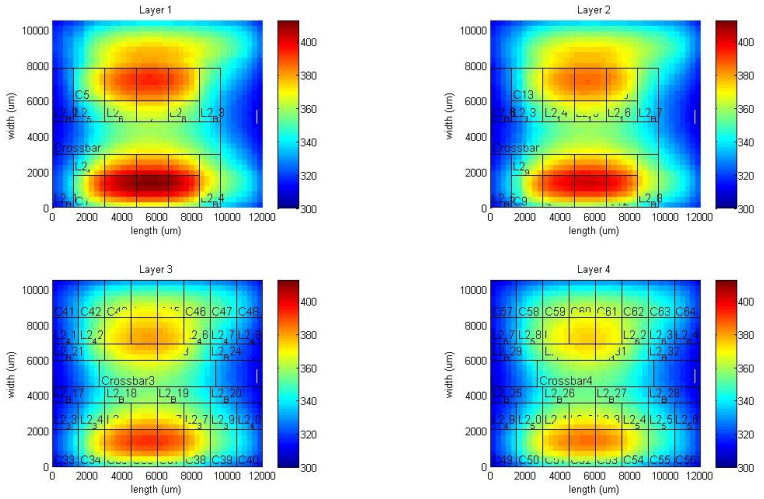


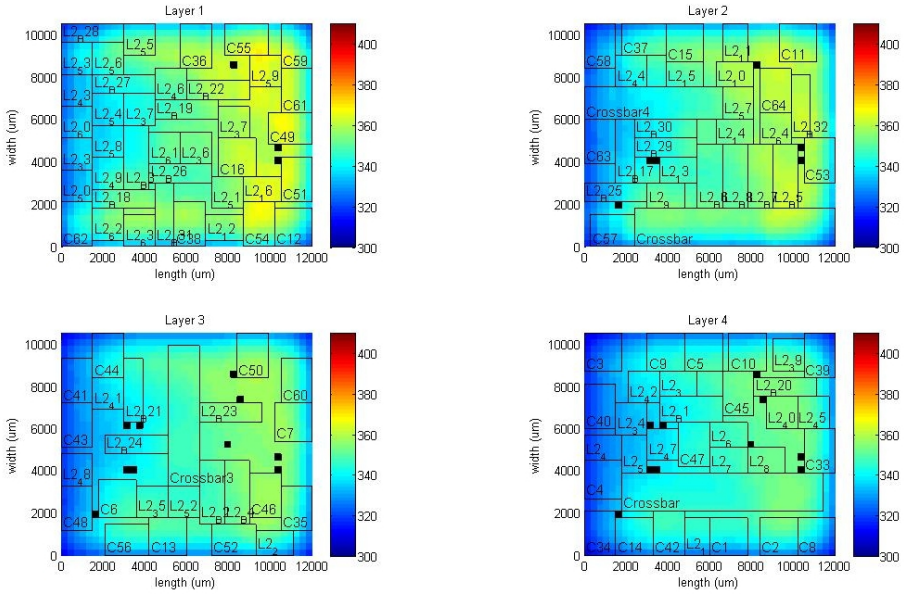
Figure 5.30: Thermal results for the original 48-core system.

optimized design did. Both thermograms are presented together in Figure 5.31.

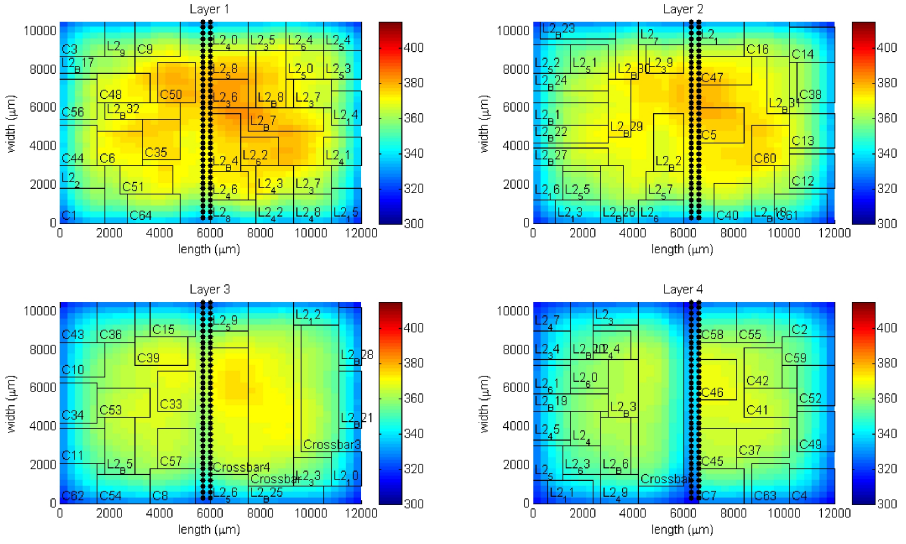
However the novel technique of adding isolating air channels has to be used in combination with other active cooling techniques as liquid μchannels, to be proven as an effective way of reducing the temperature in the 3D stack. To this end, and once the placement has been done, liquid channels can be deployed in the 3D IC.

As different thermal regions have been created, most of the optimized channels will be placed in the hot regions. On the other hand, some liquid channels will be located with heat sinks units in order to obtain an homogeneous thermal distribution.

Table 5.16 shows the results when functional units have been placed in isolated areas, and liquid channels have been also optimized to decrease the temperature in the stack. With this approach, we can save to place 12 liquid channels in order to obtain the same thermal results that were obtained before with 32. These results are shown in Figures 5.32a and 5.32b, where the differences between the two floorplans is depicted. Table 5.16 shows the thermal metrics analyzed along the chapter. These results have a very important impact in terms of technological costs and fabrication issues, and also a cost in the pumping energy required to move liquid through the channels. Since some functional units are separated by the air channels there is a small overhead in the wirelength ($<$1 %) when compared to the scenario without thermal regions. This overhead is compensated by the thermal benefits and the saving in the number of liquid channels. According to [93] 5W of operating energy for the pumping system is saved with the reduction of 12 μchannels.

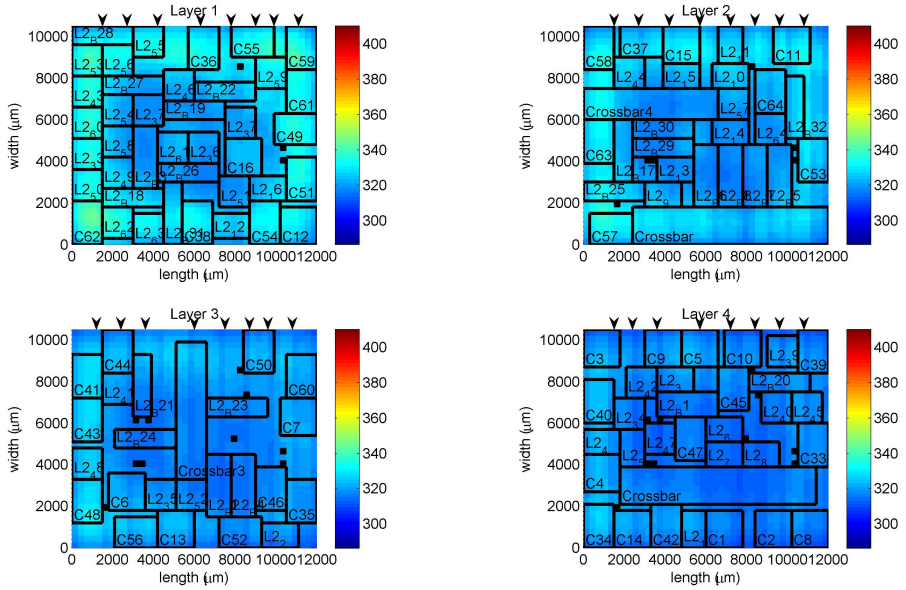


(a) Optimized 48-core system without air channels

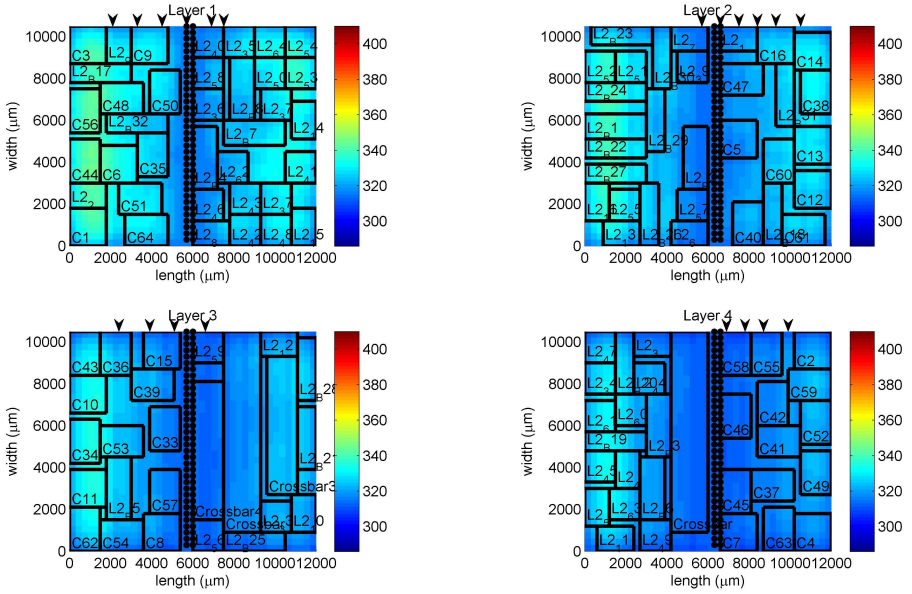


(b) Optimized 48-core system with air channels

Figure 5.31: Comparison of the optimization without and with air channels.



(a) Thermal results for the optimized 48-core system without air channel isolation and 32 liquid channels.



(b) Thermal results for the optimized 48-core system with air channel isolation and 20 liquid channels.

Figure 5.32: Comparison of the optimization without and with air channels and liquid μchannels.

	Max. T	*Mean T*	*Grad.*	*Deviation*	*Wirelength*
Air isolation without channels	377	355	61	14.6	2104
Without air isolation+32channels	335	318	24	4.5	1459
Air isolation +20 channels	335	322	27	4.5	2104

Table 5.16: Comparison between optimized floorplan with and without air channel isolation plus liquid microchannels.

Implementing this novel technique not only achieves the same thermal results when it is combined with liquid cooling but also reduces the number of necessary μchannels.

Chapter 6

Conclusions and Future Work

We know very little, and yet it is
astonishing that we know so much, and
still more astonishing that so little
knowledge can give us so much power.

Bertrand Russel

Throughout these PhD thesis it has been shown in what 3D technology consists of and its potential advantages and drawbacks. The goal of this presented work is to address the 3D thermal problem by proposing a combination of traditional cooling techniques with other novel thermal management methods.

In the following the conclusions that summarize the previous chapters will be presented as well as the references of the publications resulting from the presented work. Finally future lines of promising research will be presented.

6.1. Conclusions

3D integration has become a real important line of research since the traditional integration of electronic devices is reaching a physical limit and hence further improvements in the performance of the processors. 3D has emerge as a solution for the communication delay due to the big distances among processors, which constitute a very important bottleneck in the development of new multi core chips.

However this new way of integrating electronic devices has some drawbacks related to their reliability, which is mainly affected by the high temperatures reached inside the 3D stack which are very difficult to dissipate. This PhD thesis faces the thermal aware floorplanning problem for large 3D Multi Processor System on Chip (MPSoC).

Inner layers of the stack are not able to dissipate heat to the environment

which creates prohibited high temperatures inside the chip. These elevated temperatures affect the lifetime of the device because of the creation of hot spots or thermal gradients. Since the problem is mainly located in inner layers, traditional hardware techniques focused on the package dissipation capability such as electric fans, different packages, external liquid circuits, heat sinks... are not able to suffocate the issue. Although all these traditional methods are no longer useful by themselves they can still coexist with novel cooling techniques adapted, or specifically designed for 3D architectures. In the previously presented work several techniques that can address the 3D thermal problem have been used, however, as has been clearly stated, the need of developing automatic tools capable of attacking the thermal problem in the design phase of the device is vital for ensuring the feasibility of the 3D stack.

The presented work is based on a real multi core system (Niagara) which is thermally optimized using our proposed thermal aware floorplanner. Our floorplanner is capable of optimizing the positions of the functional units according to their power consumption, reducing the thermal impact that hot units have in the entire design volume due to their high power consumption.

A new representation for the evolutionary algorithm, that allows the placement of hot functional units far from each other, is among the enhancements presented by this thesis. With this representation, that differs from the techniques found in the literature and that have been widely used such as combined bucket, O-trees, 2D array, Polish notation... our floorplanner is able to place hot functional units far from each other and in regions where environmental dissipation is favored. Besides, this new representation ensures and allows the placement of enough TSVs to fulfill communication needs during the optimization phase. Combining these features, our floorplanner, as far as we know, is the only CAD tool that allows the optimization of different temperature control techniques such as the placement of functional units, placement of TSVs and the optimization of liquid μchannels.

Liquid μchannels have been proven to be a very efficient way of reducing temperature levels inside the 3D stack with an admissible power consumption due to the additional pumping system. Our floorplanner not only improves the results obtained by these channels when they are disposed in an homogeneous way, but also reduces the number of required μchannels which is translated in an overall better power statistic.

On the other hand, our proposed representation has been used for the developed Multi-Objective Evolutionary Algorithms which have been used throughout the optimization process. These algorithms achieve very good results when optimizing performance (not incrementing the wire length) and the maximum temperature. The algorithms return a set of solutions which conform a Pareto front. As has been well proven, MOEAs are offered as an easy and a suitable tool in order to face multi objective NP-hard problems.

Applying these set of solutions the obtained results show that the temperature is reduced below tolerable limits without drastically impacting on the performance.

On the other hand a brand new way of 3D design has been introduced, thermal domains. As has been above described, these thermal domains are mapped in the 3D stack using air channels directly etched in the silicon wafer. With the existence of these thermal regions, the designer is able to know in advance, which regions of the stack will be hotter or colder, and then, apply additional cooling techniques in the hotter areas. This revolutionary technique has proven to achieve better thermal results alleviating the deployment of liquid μchannels.

Reviewing the objectives presented in Chapter 1, this PhD thesis has achieved the following results:

1. *To show the architectural possibilities to cope with the thermal problem and the possibilities of 3D adaptation.* Chapter 2 show how the electronic industry has evolved towards 3D integration taking advantages of the wire length reduction which has come to be one major problem in integration capability. However, as has been explained, 3D integration entails thermal dissipation issues.

2. *To use a combination of hardware and design techniques that include active cooling system placement.* Chapters 3 and 4, exhibit several techniques to address the thermal problems derived from power consumption in 3D integration technology. In these chapters the traditional techniques are compared and adapted to be suitable for the 3D case. Among all the presented techniques the above presented work has used to face the thermal behavior: *selective placement of functional units, optimization of liquid μchannels, traditional heat sinks* and *fans.*

3. *To propose a thermal-aware floorplanner based on a multi objective evolutionary algorithm which is able to propose feasible 3D scenarios.* In Section 4.3, our proposed algorithm is described. This floorplanner carries out the entire thermal aware design using two different MOEAs which return a set of feasible solutions for the 3D integration scenario.

4. *To propose a novel design technique based on the creation of thermal regions to optimize the cooling effectiveness of the proposed solutions.* In Section 3.2, the revolutionary method of thermal regions, and its basis is presented. By using air channels as an isolator, the chip can the thermally managed much more efficiently as has been proven with the above presented results.

5. *To use 3D thermal simulators to validate the results delivered by the design algorithm.* The whole design process has been tested with an

adapted 3D thermal model described in 4.1. This model is based in several previous works, however it has been changed in order to include TSVs, air channels and liquid μchannels.

6.2. Future Work

This PhD thesis, has demonstrated that large and complicated 3D structures can be manufactured from the thermal point of view if thermal-aware floorplanning techniques are used during the device design phase. However, as has been claimed several times, 3D integration technology is taking its first steps, and therefore there are many researching areas with a promising future. Following the structure that has been presented along the text, they can be divided in three main lines, architectural solutions, thermal aware design techniques and specialized software for 3D processing stacks.

Architectural solutions must be proposed not only for the thermal aware design, buy also to improve 3D integration manufacturing flow. From the silicon wafer, many industrial processes, as shown in Chapter 2, must be done carefully, since every step directly impacts on the entire 3D stack. Achieving good results in the manufacturing processes is the key for the develop of 3D technology.

As has been explained, communication plays a very important role. It is becoming the bottleneck for the traditional 2D systems, but the problem will be transferred to the 3D scenario if no new strategies are used. This has open a very important topic of research among the 3D integration researchers. Communication in large MPSoCs is moving towards the implementation of Network on Chip (NOC), and optimized and complicated crossbars as the ones used in previous tests extracted from the real Niagara design. NOCs offer a very attractive and effective solution for the communication problem in large MPSoCs but its thermal contribution to the whole electronic 3D IC must be also taken into account.

Another very important factor that affects the communication is the existence of TSVs. It has been proven that TSVs offer a better solution in terms of reliability, efficiency and costs, than wire bonding. The industry has already made great advantages in the manufacturing process however many improvements in their fabrication have to be done. Although this work has not considered the deployment of TSVs with thermal purposes (remember TTSVs explained in Section 3.1.3.2), TTSVs could be used as an additional technique to the ones proposed by this PhD thesis.

Active cooling techniques are quite new. In the above presented results, liquid μchannels have proven to be a very efficient method to alleviate the temperature of inner layers of the 3D stack. Our work has shown how the μchannels work with a constant flow rate, however, adapting the speed of the coolant to the thermal conditions of the 3D chip, can reduce the power

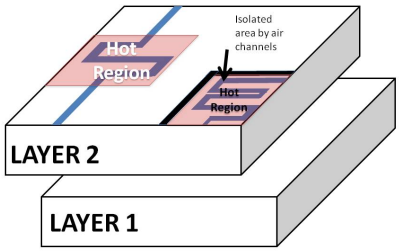


Figure 6.1: Alternative shapes for optimizing liquid μchannels.

consumption of the pumping, resulting in less energy costs of the overall system. The flow rate of the μchannels could be dynamically adapted to special situations or transient thermal scenarios, reducing pumping costs improving the thermal footage. In this line, direction of the μchannels could be also optimized. In our study we have only considered channels from south to north, and other directions or shapes could be taken into account as depicted in the example Figure 6.1.

The technique of creating thermal regions first proposed in this PhD thesis opens a new line of investigation itself. The study carried out above, shows that promising results in terms of thermal behavior can be achieved if the regions are wisely defined. Obviously further improvements and tests must be done variating the area and the locations of the different thermal areas.

On the other hand, as described in Chapter 4, the architecture and design techniques have to be optimized by different CAD tools that allow to take full advantage of them. This work has presented a new representation for the entire 3D floorplan, that allows, not only to describe the position of the functional units throughout the chip, but also the placement of TSVs when they are not considered as a functional unit. However the developed of the algorithm is susceptible of being improved by parallelization. The proposed representation has proven to be better than other state of the art traditionally used as Combined Bucket and 2D Array which have been developed using Simulated Annealing algorithms.

Finally, several improvements in software techniques for thermal purposes can be easily risen. This thesis has taken a worst case scenario to prove the necessity of thermal aware design techniques and their efficiency in cooling 3D stacks, however worst case is seldom found. Studying the dynamic behavior of the 3D stack could entail a reduction in unnecessary cooling systems. From the traditional operative system techniques used to migrate task among cores as the one presented in Section 4.2.4, to the creation of voltage and frequency islands, the research area covers many fields of Architecture

and Technology of Computing Systems. Profiling the thermal behavior and power consumption of the functional units of the 3D stack would turn in in more information that can be transferred to the optimization tools in order to achieve better results using less resources.

6.3. Publications

Here there is a list with the papers published during my research:

6.3.1. Publications in International Conferences

- D. Cuesta, J. L. Ayala, J. I. Hidalgo, D. Atienza, A. Acquaviva and E. Macii. «Adaptive Task Migration Policies for Thermal Control in MPSoCs». In Proc. *International Symposium on Very Large Scale Integration (ISVLSI)*. 2010, pp. 110-115.

- D. Cuesta, J.L. Risco and J.L. Ayala. «3D Thermal-Aware Floorplanner for Many-Core Single-Chip Systems». In Proc. *Latin American Test Workshop (LATW)*. 2011, pp. 1-6.

- D. Cuesta, J. L. Ayala, J. I. Hidalgo, M. Poncino, A. Acquaviva and Enrico Macii. «Thermal-aware floorplanning exploration for 3D multi-core architectures». In proc. *Great Lakes Symposium on VLSI (GLSVLSI)*. 2010 pp. 99-102.

- D. Cuesta, J.L. Risco and J.L. Ayala. «3D Thermal-Aware Floorplanner using a MILP Approximation». In Proc. *Design of Circuits and Integrated Systems (DCIS)*. 2010.

- D. Cuesta, J.L. Risco, J.L. Ayala and J.I. Hidalgo. «A combination of evolutionary algorithm and mathematical programming for the 3D thermal-aware floorplanning problem.». In Proc. *Genetic and Evolutionary Computation Conference (GECCO)*. 2011, pp. 1731-1738.

6.3.2. Publications in International Journals

- D. Cuesta, J.L. Risco-Martin and J.L. Ayala. «3D thermal-aware floorplanner using a MILP approximation». In Journal *Microprocessors and Microsystems*. 2012. Vol. 36, pp 344-354.

- D. Cuesta, J.L. Risco-Martin, J.L. Ayala and J.I. Hidalgo. «3D thermal-aware floorplanner using a MOEA approximation». In Journal *Integration, the VLSI Journal*. 2012, Vol. 46, pp. 10-21.

- J.L. Ayala, A. Sridhar and David Cuesta. «Thermal modeling and analysis of 3D multi-processor chips». In Journal *Integration -Amsterdam*. 2010. Vol. 43, pp. 1-15.

Appendix A

Optimization Methods

Si se puede imaginar ... se puede programar :-)

Anónimo

This appendix slightly describes the basis and the mathematical principles of the most used techniques in the thermal optimization processes that can be found in the literature. Among these techniques Linear Programming, Simulated Annealing and Evolving Algorithms are the most important.

A.1. Linear Programming (LP)

Linear programming is a mathematical method to calculate the best fit (maximum or minimum depending on the problem) in a mathematical model that must be formulated using linear relationships. Mathematical optimization is a wide field of investigation and Linear Programming is just a specific case.

In order to be able to solve an optimization problem using LP, this must be formulated with linear objective functions, that means, with linear equalities and linear inequalities constraints. The solutions found by this method must be inside a convex space of search defined by linear inequalities as it is shown in Figure A.1.

The objective function must be real, and of course it has to be defined inside the space of search. LP technique, is developed to find a point inside the space of search that minimizes or maximizes the objective function.

In general, linear problems that can be expressed in this canonical form:

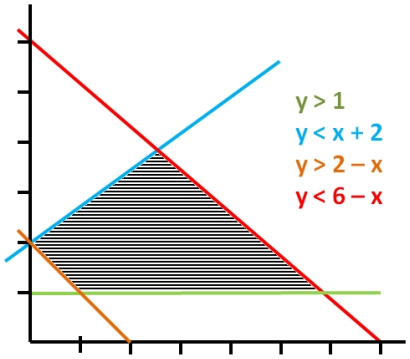


Figure A.1: Example of a Search Space defined by some inequalities.

$$\text{maximize} \quad \mathbf{c}^{\mathrm{T}}\mathbf{x} \tag{A.1}$$

$$\text{subject to} \quad A\mathbf{x} \leq \mathbf{b} \tag{A.2}$$

$$\text{and} \quad \mathbf{x} \geq \mathbf{0} \tag{A.3}$$

where x expresses the solutions that must be found by solving the equation system. c and b are known coefficients to the problem. A is also a known matrix of coefficients, and (c^{T} is the transpose of the matrix. The objective function to be optimized is equation A.1. The search space is defined by a series of constraints defined in A.3. A very simple example of this search space can be seen in Figure A.1.

Linear programming can be applied to a lot of problems in very different fields such as logistics ([50]), nuclear physics ([95]), medicine ([110]), economics ([126]) or computer science ([65]). For more detail about LP the reader is referred to [106].

When a linear problem is discrete and the solutions are integers we are talking about a Integer Linear Problem (ILP) problem. ILP problems can be solved in polynomial time. In practice linear problems can be solved efficiently for reasonable-sized problems, or even for big problems if they are well formulated and structured. When some of the variables are not integer, the problem becomes Mixed Integer Linear Problem (MILP).

The most common algorithms to solve a ILP problem are Simplex and its derivations. Simplex method was invented by George Dantzig in 1947, and it was listed as one of the top ten algorithms of the twentieth century by the journal of Computing in Science and Engineering ([78]). The method tests adjacent vertex of the feasible solution set in sequence. At every new vertex the objective function must improve or stay unchanged.

The simplex method is very efficient and it takes 2η or 3η, where η is the number of inequalities constraints to get the optimum solution. Simplex

method has been improved and adapted to special problems and formulations by many authors. Dantzig wrote the principles of linear programming and the simplex method in [39].

Based on Simplex, different type of algorithms have been developed as interior point method. The complexity of this algorithm is polynomial. The algorithm builds a set of solutions that converge to the optimal. The first author was Karmarkar ([86]. This algorithm was later modified by Mehrotra ([107])to solve large-scale problems, by a predict-correct method which guides the algorithm towards the solution much faster.

A.2. Simulated Annealing

Simulated annealing (SA) is a probabilistic metaheuristic method for finding optimum solutions. This mathematical method was first described in 1983 by Scott Kirkpatrick, C. Daniel Gelatt and Mario P. Vecchi in [89]. The method was born as a modification of the MonteCarlo technique, based on the search through random solutions. In the paper, the authors solve a problem related to the placement of physical elements in a computer and compare its performance with other algorithms, using the travel salesman problem, offering a better convergence time than previous algorithms.

SA locates a good approximation to the global optimum in the entire search space. SA works as an iterative method which tries to replace the current found solution by another random one. This random candidate is constructed following certain constraints or distributions, that have to be added to the algorithm, depending on the nature of the problem. The new candidate is then evaluated and compared to the last solution. The new solution may then be accepted with a probability that depends both, on the difference between the corresponding function values, and also on a global parameter T (called temperature). T is decreased during the optimization problem. This principle is from the manufacturing process from the name is taken. Annealing is a metallurgic process that consist on warming up a metal and later letting it slowly cool down. T becomes low in the mathematical process and focus the solutions to the optimum.

The ability of escaping from local optimums is what confers this algorithm its strength. SA achieves this using two mathematical tricks. The first one consists of an algorithm developed by Metropolis et Al. in 1953 called "Metropolis Algorithm"([108]). This algorithm saves not optimum or better solutions based on a probability given by the Equation (A.4).

$$e^{(-\Delta D/T)} > R(0,1), \tag{A.4}$$

Combining the parameters in the previous equation, the probability of choosing new good or bad solutions changes with time. ΔD is the change of

the new solutions compared to the previous one. It is negative when the new solution is better, and positive when the new solution offers worse results. R is a random number between 0 and 1. D is called cost function. In the particular case of metal annealing, it would be the free energy of the material, and it would actually be k_BT, where k_B is the Boltzman's constant and T is the physical temperature of the metal. In the equation, T plays a very important role, and the dependency of the algorithm with this parameter is such that, if T is large, the probability of accepting "bad" results will be high and, as T decreases, the solutions are better and better which is the second trick of the algorithm. After lowering the temperature several times to a low value, one may then break the process by accepting only "good"solutions in order to find the local minimum of the cost function. There are many different studies for lowering the temperature, these methods lead the algorithm differently to the final solution and are called "annealing schedule" techniques.

A.3. Evolutionary Algorithms (EA)

Evolutionary computation, has its origin in the 50s and 60s when some independent researchers started to study evolutionary systems in the nature, based on the theory developed in 1859 by Charles Darwin [40]. They realized that these natural phenomena could be applied mathematically, optimizing engineering problems following the principle "as nature evolves, specimens have to adapt themselves to new environments". These natural methods were then modeled, and mathematical operators, based on natural selection and mutation were raised as the base of the evolutionary algorithms.

We will stress the importance of EAs because our work will be inspired in this technique. It was Bagley in 1967 who used the term Genetic Algorithms (GA)for first time ([10]). In his PhD Thesis, Bagley designed some genetic algorithms to search parameters sets to evaluate games. However, it is John Holland, the one considered as the father of the genetic algorithms. Opposite of evolutionary strategies, and evolutionary programming, the goal of Holland was studying, in a formal way, the natural adaptation mechanisms, instead of applying the algorithms to solve specific problems. It is in [70], where Holland sets the basis of the GA, presenting it as an abstraction of the biological evolution, and presenting the theoretical methods of mutation, selection and crossover.

In the 90s and nowadays, the border between evolutionary algorithms and genetic algorithms is not as clear as was before, and the terms, evolutionary algorithm or genetic algorithm are used indistinctly. The scope of these algorithms is much wider than the term used by Holland.

In the beginning the objectives that Holland and his colleagues pursued were two. The first one consisted of abstracting and explaining the adaptive process of natural systems, and the second was designing artificial systems

Nature	**Genetic Algorithm**
Environment	Optimization problem
Individuals that live in the environment	Feasible solution
Individual's degree of adaptation to the environment	Solutions quality (fitness function)
Species	A set of solutions
Selection, recombination and mutation	Mathematical operators
Evolution of populations	Applying mathematical operators to a set of solutions

Table A.1: Metaphor between nature and GA

that could behave as natural systems do. In this way, as Goldberg said in [60]: "Genetic algorithms are search algorithms based on the mechanics of natural selection and natural genetics. They combine survival of the fittest among string structures with a structured yet randomized information exchange to form a search algorithm with some of the innovative flair of human search."

In Table A.1, the most important features of evolution metaphors between nature and GAs are showed.

Although GA will be later explained in more detail, basically they work, as nature does, evolving generations. In every generation a new set of individuals (solutions of the optimization problem) is created, then some of the individuals that compose the population are chosen to create, in the next generation, the new set of solutions. The novelty that GAs add is that they exploit efficiently historic information to obtain new better solutions.

Maybe, the most important questions are, Why do we use evolution as an inspiration for solving real problems? and, What can we extract from natural processes? These questions can be answered when we realize that there exist a huge number of possible solutions in such a huge system as nature. The optimization problems formulated using GA can take advantage of parallel computation, where many possibilities are studied and explored simultaneously. In fact, nature, uses a method of searching among a huge number of possible "solutions" for the problem, these solutions are new specimens of the new generation. As can be imagined, the possibilities of feasible solutions for these new population are enormous, combining all the possible genetic sequences. The goodness of the solutions will be tested empirically if the organisms are able to survive and reproduce in their environments.

Natural conditions are constantly changing which means that every new generation has to adapt to the new conditions if they want to survive. This is precisely what adaptive computation requires. Furthermore, natural evolu-

tion is a massively parallel search method, instead of working on a specimen, nature tests and changes millions of organisms in parallel, choosing as Darwin postulated, those individuals that are more prepared to survive under new conditions.

Summarizing, the rules of evolution can be described as set of random mechanisms that evolve the population via mutations, genetic recombination and then followed by a natural selection method.

To a better understanding of the mathematical methods and evolutionary operators some biological terminology, that will be then used, must be introduced.

Living organisms are made by cells, which constitute the base of the organic matter. Each of those cells contains the same set of one or more *chromosome* strings, which compose the DNA, the fingerprint of any biological organism.

Each chromosome, can be divided into *genes*. These genes encode a particular protein, which can be understood as a certain feature of the individual such as the color of the eyes or the color of the hair. As can be deduced, these features can take different values. The color of the eyes of an individual can be blue, brown, green... this set of values are given by the *alleles*.

One of the most important part that must be taken into account is the position of the genes. Every gene is located at a particular position in the chromosome, which fixes the encoding of the full genetic material. When all the chromosomes are considered together, we can refer to the complete collection of genetic material, and we would be talking about the organism's genome.

The term *genotype* refers to the particular set of genes contained in a genome. The genotype gives all the characteristics of an individual.

In the following we will present how reproduction occurs, and its relation with GAs. In nature there are two kinds of organisms (genetically speaking). Those organisms that have a single chromosome are called haploid, and those that have their genetic information in a pair of chromosomes are called diploid. Diploid organisms, during sexual reproduction, recombine or make a crossover of their chromosomes. Each parent contributes with their genes, and these genes are exchanged between each pair of chromosomes to form a full new chromosome. Then this chromosome pairs up with another set of genes to create a full set of diploid chromosomes.

On the other hand, haploid organisms (most commonly used in GAs studies), during the reproduction process, exchange their genes between the two parents'single chromosomes. In this process, offspring are subject to mutation, because single elements of the DNA are changed from parent to offspring. In nature these changes result from genetic copying errors. This fact leads us to a new term, *fertility*. Fertility is defined as the probability that the offspring will live depending on the number of them.

The relationship between natural process and GAs comes by the hand of the term chromosome. In GAs this term refers to a candidate solution to the problem. One of the most important issues, when describing an optimization problem that will be solved using GAs, is the codification of the chromosome. Tipically, chromosomes are encoded as a bit string. If chromosomes are encoded, as it was just said, as a bit string, then each bit, or a set of them, in the string would be a gene. In our example, the allele is either, 0 or 1. Continuing with our example. The configuration of bits in the individual's chromosome is called genotype.

Once all the biological terms have been explained, the reader could imagine how a GA works. An overview of a simple genetic algorithm is presented in the pseudocode given in Algorithm A.1.

In the process of the simple genetic algorithm, there are, at least, three operators: selection, crossover and mutation. Every operator has many ways of being applied, and many researches have been conducted in this field. Obviously this is not the scope of this study and only an overview of these operators will be shown.

Selection. This operator selects among all, some chromosomes in the population to initiate the reproduction. There are many selection mechanisms in the literature such as the roulette method or the tournament. Selection is based on choosing those chromosomes which better fit the solution, what implies that those considered as better solutions will be chosen more often to reproduce.

Crossover. In the crossover, or recombination, a gene, or a random set of them is chosen and exchanged to create different offspring. For example, we have two different chromosomes, 10110110 and 11100011. The crossover operator can exchange the set of the first four genes creating two new and different offspring, **1110**0110 and **1011**0011.

Mutation. Mutation consists of adding some random changes in the chromosome. Continuing with our binary example mutation could change

Algorithm A.1 Simple Genetic Algorithm

```
1: Produce an initial population of individuals
2: Evaluate the fitness of all individuals
3: while termination condition not met do
4:     select fitter individuals for reproduction
5:     recombine between individuals
6:     mutate individuals
7:     evaluate the fitness of the modified individuals
8:     generate a new population
9: end while
10: return Final Population
```

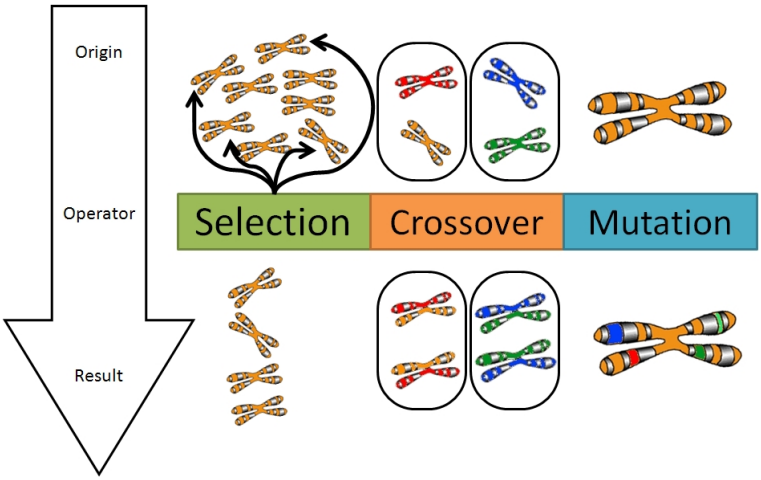


Figure A.2: Example of the three most important genetic operators.

the chromosome 10110110. This solution could be randomly mutated by flipping one or some bits generating a new solution 10**0**10**0**11. As it happens in nature, mutation can occur in every position, but the probability is very low. Normally, this probability is set in the order of $1 \cdot 10^{-3}$.

A graphical scheme of how these operators work is shown in Figure A.2, where different diploid chromosomes are the inputs to the genetic operators.

Each operator has a different impact on the algorithm, and the usage of all three operators makes GAs a powerful optimization tool. Using selection alone, will then fill the population with copies of the best individuals from the initial population, but the algorithm will never find an optimum solution if it is not among the initial population, because there will not be new genetic members. On the other hand, when selection is combined with crossover, the algorithm will tend to converge on a good but not optimal solution, because new solutions can only be given by a combination of the initial individuals. If mutation is used alone, the algorithm will search in the entire space of solutions, but it will not be focused on optimal solutions. Using selection and mutation creates a parallel, noise-tolerant, hill climbing algorithm. As it was described in Algorithm A.1, is the combination of all three operators what guides the GA to find optimum solutions.

Bibliografía

El experimentador que no sabe lo que está buscando no comprenderá lo que encuentra.

Claude Bernard

[1] http://www.opensparc.net/pubs/preszo/07/n2isscc.pdf.

[2] H. Abbas. Accurate resolution of signals using integer-coded genetic algorithms. In *Evolutionary Computation, 2006. CEC 2006. IEEE Congress on*, pages 2888 –2895, 0-0 2006.

[3] A. Alnukari, P. Guillemet, Y. Scudeller, and S. Toutain. Active heat-sink antenna for radio-frequency transmitter. *Advanced Packaging, IEEE Transactions on*, 33(1):139 –146, feb. 2010.

[4] M. Arik and A. Bar-Cohen. Immersion cooling of high heat flux microelectronics with dielectric liquids. In *Advanced Packaging Materials, 1998. Proceedings. 1998 4th International Symposium on*, pages 229 –247, mar 1998.

[5] I. Arnaldo, A. Cuesta-Infante, J. Manuel Colmenar, J. L. Risco-Martin, and J. L. Ayala. Boosting the 3d thermal-aware floorplanning problem through a master-worker parallel moea. *Concurrency and Computation: Practice and Experience*, 25(8):1089–1103, 2013.

[6] I. Arnaldo, J. Risco-Martin, J. L. Ayala, and J. I. Hidalgo. Power profiling-guided floorplanner for 3d multi-processor systems-on-chip. *Circuits, Devices Systems, IET*, 6(5):322–329, 2012.

[7] D. e. A. Atienza. HW-SW emulation framework for temperature-aware design in MPSoCs. *ACM Trans. Des. Autom. Electron. Syst.*, 12(3):1–26, 2007.

[8] J. L. Ayala, A. Sridhar, and D. Cuesta. Thermal modeling and analysis of 3D multi-processor chips. *Integration -Amsterdam-*, 43(7):1–15, 2010.

[9] J. L. Ayala, A. Sridhar, V. Pangracious, D. Atienza, and Y. Leblebici. Through silicon via-based grid for thermal control in 3D chips. In *Nanonet*, 2009.

[10] J. D. Bagley. *The behavior of adaptive systems which employ genetic and correlation algorithms.* PhD thesis, Ann Arbor, MI, USA, 1967. AAI6807556.

[11] J. Berntsson and M. Tang. A slicing structure representation for the multi-layer floorplan layout problem. In *EvoWorkshops*, pages 188–197, 2004.

[12] S. Bertozzi, A. Acquaviva, D. Bertozzi, and A. Poggiali. Supporting task migration in multi-processor systems-on-chip: a feasibility study. In *Proceedings of the conference on DATE*, pages 15–20, 2006.

[13] B. Black, D. Nelson, C. Webb, and N. Samra. 3d processing technology and its impact on ia32 microprocessors. In *Computer Design: VLSI in Computers and Processors, 2004. ICCD 2004. Proceedings. IEEE International Conference on*, pages 316–318, 2004.

[14] G. Boogle. *A Treatise on the Calculus of Finite Differences, 2nd Editioni.* Dover, 1960.

[15] C. Brench. Heatsink radiation as a function of geometry. In *Electromagnetic Compatibility, 1994. Symposium Record. Compatibility in the Loop., IEEE International Symposium on*, pages 105 –109, aug 1994.

[16] E. W. Brião, D. Barcelos, F. Wronski, and F. R. Wagner. Impact of task migration in NoC-based MPSoCs for soft real-time applications. In *Proceedings of the International Conference on VLSI*, pages 296–299, 2007.

[17] J. Brownlee. *Clever Algorithms: Nature-Inspired Programming Recipes.* Lulu.com, 2011.

[18] F. Carson, H. T. Lee, J. H. Yee, J. Punzalan, and E. Fontanilla. Die to die copper wire bonding enabling low cost 3D packaging. In *Electronic Components and Technology Conference (ECTC)*, pages 1502–1507, 2011.

[19] H. W. D. Chang and W. J. B. Oldham. Dynamic task allocation models for large distributed computing systems. *IEEE Transactions on Parallel Distributed computing Systems*, 6:1301–1315, 1995.

[20] Y.-W. Chang, T.-C. Chen, and H.-Y. Chen. Physical design for system-on-a-chip. In Y.-L. Lin, editor, *Essential Issues in SOC Design*, pages 311–403. Springer Netherlands, 2006.

[21] G. Chen and S. Sapanetkar. Partition-driven standard cell thermal placement. In *International Symposium on Physical Design*, 2003.

[22] Y.-X. Chen, Y.-J. Huang, and J.-F. Li. Test cost optimization technique for the pre-bond test of 3D ICs. In *VLSI Test Symposium (VTS), 2012 IEEE 30th*, pages 102–107, 2012.

[23] T. Ching-Han and K. Sung-Mo. Standard cell placement for even on-chip thermal distribution. In *Proceedings of the 1999 international symposium on Physical design*, ISPD '99, pages 179–184, New York, NY, USA, 1999. ACM.

[24] K. Choi, K. Dantu, W.-C. Cheng, and M. Pedram. Frame-based dynamic voltage and frequency scaling for a MPEG decoder. In *Proceedings of the 2002 IEEE/ACM international conference on Computer-aided design*, ICCAD '02, pages 732–737, New York, NY, USA, 2002. ACM.

[25] D. Choudhury, J. Foschaar, and D. Rensch. A method of heatsink fabrication for millimeter wave high-power gunn devices. In *Electronics Manufacturing Technology Symposium, 1998. Twenty-Third IEEE/CPMT*, pages 386 –389, oct 1998.

[26] C. C. N. Chu and D. F. Wong. A matrix synthesis approach to thermal placement. In *International Symposium on Physical Design*, 1997.

[27] C. Coello, D. A. Van Veldhuizen, and G. B. Lamont. *Evolutionary Algorithms for Solving Multi-Objective Problems*. Kluwer, 2002.

[28] J. Cong, G. Luo, J. Wei, and Y. Zhang. Thermal-aware 3D IC placement via transformation. In *Design Automation Conference, 2007. ASP-DAC '07. Asia and South Pacific*, pages 780 –785, jan. 2007.

[29] J. Cong, J. Wei, and Y. Zhang. A thermal-driven floorplanning algorithm for 3D ICs. In *Proceedings of the 2004 IEEE/ACM International conference on Computer-aided design*, ICCAD '04, pages 306–313, Washington, DC, USA, 2004. IEEE Computer Society.

[30] J. Cong and Y. Zhang. Thermal via planning for 3-D ICs. In *Computer-Aided Design, 2005. ICCAD-2005. IEEE/ACM International Conference on*, pages 745 – 752, nov. 2005.

[31] A. Coskun, J. Ayala, D. Atienza, and T. Rosing. Modeling and dynamic management of 3D multicore systems with liquid cooling. In *Very Large Scale Integration (VLSI-SoC), 2009 17th IFIP International Conference on*, pages 35–40, 2009.

[32] A. Coskun, J. Ayala, D. Atienza, T. Rosing, and Y. Leblicini. Dynamic thermal management in 3D multicore architectures. *Design, Automation and Test in Europe*, pages 1410–1415, 2009.

[33] L. Covert, J. Lin, D. Janning, and T. Dalrymple. Dual-function 3-D heatsink antenna for high-density 3-D integration. In *Radio-Frequency Integration Technology, 2007. RFIT 007. IEEE International Workshop on*, pages 26 –29, dec. 2007.

[34] D. Cuesta, J. L. Ayala, J. Hidalgo, D. Atienza, A. Acquaviva, and E. Macii. Adaptive task migration policies for thermal control in MPSoCs. In *ISVLSI*, pages 110–115, 2010.

[35] D. Cuesta, J. L. Ayala, J. Hidalgo, M. Poncino, A. Acquaviva, and E. Macii. Thermal-aware floorplanning exploration for 3D multi-core architectures. In *GLSVLSI*, pages 99–102, 2010.

[36] D. Cuesta, J. L. Risco-Martin, and J. L. Ayala. 3d thermal-aware floorplanner using a MILP approximation. *Microprocessors and Microsystems*, 36(5):344 – 354, 2012.

[37] D. Cuesta, J. L. Risco-Martin, J. L. Ayala, and J. I. Hidalgo. 3d thermal-aware floorplanner using a MOEA approximation. *Integration, the VLSI Journal*, 2012.

[38] W. Dally. Stream processors: Programmability with efficiency. *ACM Queue*, 2(1):52–62, 2004.

[39] G. B. Dantzig. *Linear Programming and Extensions.* Princeton University Press, 1963.

[40] C. Darwin. *On the Origin of Species by Means of Natural Selection; or, the Preservation of flavored Races in the Struggle for Life.* 1859.

[41] S. Das, A. Chandrakasan, and R. Reif. Calibration of Rent's rule models for three-dimensional integrated circuits. *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, 12(4):359–366, 2004.

[42] T. A. Davis and I. S. Duff. An usymmetric-pattern multifrontal method for sparse lu factorization. *SIAM Journal on Matrix Analysis and Applications*, 18(1):140–158, 1997.

[43] J. De la Llave and M. J.A. *Diseño y Construcción de un horno para tejas de barro con una capacidad de 18,000 tejas por semana.* PhD thesis, 1997.

[44] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197, 2002.

[45] Y. Development. Yole development. 3D IC report. `http://www.yole.fr`, 2007.

[46] J. Donald and M. Martonosi. Techniques for multicore thermal management: Classification and new exploration. In *Proceedings of the 33rd international symposium on Computer Architecture*, pages 78–88, 2006.

[47] G. Druais, P. Ancey, L. Chapelon, J. Charbonnier, S. Cheramy, Y. Dodo, A. Farcy, G. Garnier, Y. Guillou, D. Henry, P. Leduc, J. Pruvost, E. Saugier, and N. Sillon. TSV as an alternative to wire bonding for a wireless industrial product: another step towards 3D integration. In *Electronic System-Integration Technology Conference (ESTC)*, pages 1–4, 2010.

[48] M. Ekpanyapong, M. B. Healy, C. S. Ballapuram, S. K. Lim, and H. hsin S. Lee. Thermal-aware 3D microarchitectural floorplanning. Technical report, Georgia Institute of Technology Center for, 2004.

[49] EPFL. 3D-ICE. `http://esl.epfl.ch/3d-ice.html`, 2012.

[50] A. L. Erera, J. C. Morales, and M. Savelsbergh. Robust optimization for empty repositioning problems. *Oper. Res.*, 57(2):468–483, Mar. 2009.

[51] A. W. et Al. Bandwidth optimization for through silicon via (TSV) bundles in 3D integrated circuits. In *Design, Automation and Test in Europe*, 2009.

[52] J. Fan and C. S. Tan. Low temperature wafer-level metal thermocompression bonding technology for 3D integration, metallurgy. *Metallurgy - Advances in Materials and Processes*, 2012.

[53] E. Farsad, S. Abbasi, M. Zabihi, and J. Sabbaghzadeh. Thermal analysis of 300W-QCW diode laser with copper foam heatsink. In *Photonics and Optoelectronics (SOPO), 2011 Symposium on*, pages 1 –4, may 2011.

[54] G. Foundries. Global foundries. `http://www.globalfoundries.com`.

[55] K. Fujiyoshi, H. Kawai, and K. Ishihara. A tree based novel representation for 3d-block packing. *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, 28(5):759–764, 2009.

[56] A. Fukunaga and S. Tazoe. Combining multiple representations in a genetic algorithm for the multiple knapsack problem. In *Evolutionary Computation, 2009. CEC '09. IEEE Congress on*, pages 2423 –2430, may 2009.

[57] J. Gascon-Moreno, S. Salcedo-Sanz, E. Ortiz-Garcia, L. Carro-Calvo, B. Saavedra-Moreno, and J. Portilla-Figueras. A binary-encoded

tabu-list genetic algorithm for fast support vector regression hyper-parameters tuning. In *Intelligent Systems Design and Applications (ISDA), 2011 11th International Conference on*, pages 1253 –1257, nov. 2011.

[58] P. Giovannini, G. Beanato, A. Cevrero, P. Athanasopoulos, and Y. Leblebici. 3D stacked multi-core processor platform with improved testability. `http://lsm.epfl.ch`.

[59] E. Girczyc. *Development of an experimental and analytical model of an active cooling method for high-power three-dimensional integrated circuit (3D-IC) utilizing multidimensional configured thermoelectric modules.* PhD, The University of Texas at Arlington, 2011.

[60] D. E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning.* Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1st edition, 1989.

[61] S. Golestani and M. Taddayon. An improved integer coded genetic algorithm for power generation dispatch problem. In *Transmission Distribution Conference Exposition: Asia and Pacific, 2009*, pages 1 –4, oct. 2009.

[62] M. Gomaa, M. D. Powell, and T. N. Vijaykumar. Heat-and-run: leveraging SMT and CMP to manage power density through the operating system. In *Proceedings of the 11th international conference on Architectural support for programming languages and operating systems*, pages 260–270, 2004.

[63] M. Götz, F. Dittmann, and T. Xie. Dynamic relocation of hybrid tasks: Strategies and methodologies. *Microprocess. Microsyst.*, 33(1):81–90, Feb. 2009.

[64] S. Govind Singh and C. S. Tan. Thermal mitigation using thermal through silicon via (TTSV) in 3-D ICs. In *Microsystems, Packaging, Assembly and Circuits Technology Conference, 2009. IMPACT 2009. 4th International*, pages 182 –185, oct. 2009.

[65] M. R. Guthaus, G. Wilke, and R. Reis. Non-uniform clock mesh optimization with linear programming buffer insertion. In *Proceedings of the 47th Design Automation Conference*, DAC '10, pages 74–79, New York, NY, USA, 2010. ACM.

[66] X. Han, V. Everett, Y. Wang, and L. Zhu. Numerical analysis of direct liquid-immersed solar cell cooling of a linear concentrating photovoltaic receiver. In *Photovoltaic Specialists Conference (PVSC), 2010 35th IEEE*, pages 003033 –003038, june 2010.

[67] M. Healy, M. Vittes, M. Ekpanyapong, C. S. Ballapuram, S. K. Lim, H. H. S. Lee, and G. H. Loh. Multiobjective microarchitectural floorplanning for 2-D and 3-D ICs. *Trans. Comp.-Aided Des. Integ. Cir. Sys.*, 26(1):38–52, Jan. 2007.

[68] S. Heo, K. Barr, and K. Asanovic. Reducing power density through activity migration. *Proceeding of the ISPD*, 2003.

[69] A. Heryanto, K. L. Pey, Y. Lim, W. Liu, J. Wei, N. Raghavan, J. Tan, and D. K. Sohn. Study of stress migration and electromigration interaction in copper/low-k; interconnects. In *Reliability Physics Symposium (IRPS), 2010 IEEE International*, pages 586–590, 2010.

[70] J. H. Holland. *Adaptation in Natural and Artificial Systems.* 1975.

[71] X. Hong, G. Huang, Y. Cai, J. Gu, S. Dong, C.-K. Cheng, and J. Gu. Corner block list: an effective and efficient topological representation of non-slicing floorplan. In *Computer Aided Design, 2000. ICCAD-2000. IEEE/ACM International Conference on*, pages 8–12, 2000.

[72] W. Huang, S. Ghosh, K. Sankaranarayanan, K. Skadron, and M. R. Stan. Hotspot: Thermal modeling for CMOS VLSI systems. *IEEE Transactions on Component Packaging and Manufacturing Technology*, 2005.

[73] W. Huang, K. Sankaranarayanan, R. J. Ribando, M. R. Stan, and K. Skadron. Accurate, pre-RTL temperature-aware processor design using a parameterized, geometric thermal model. *IEEE Transactions on Computers*, 9:1277–1288, 2008.

[74] K. Hummler, L. Smith, R. Caramto, R. Edgeworth, S. Olson, D. Pascual, J. Qureshi, A. Rudack, R. Quon, and S. Arkalgud. On the technology and ecosystem of 3d / tsv manufacturing. In *Advanced Semiconductor Manufacturing Conference (ASMC), 2011 22nd Annual IEEE/SEMI*, pages 1–6, 2011.

[75] W.-L. Hung, G. Link, Y. Xie, N. Vijaykrishnan, and M. Irwin. Interconnect and thermal-aware floorplanning for 3D microprocessors. In *Quality Electronic Design, 2006. ISQED '06. 7th International Symposium on*, pages 6 pp. –104, march 2006.

[76] W.-L. Hung, Y. Xie, N. Vijaykrishnan, C. Addo-Quaye, T. Theocharides, and M. Irwin. Thermal-aware floorplanning using genetic algorithms. In *Quality of Electronic Design, 2005. ISQED 2005. Sixth International Symposium on*, pages 634 – 639, march 2005.

[77] IBM. IBM. `http://www.IBM.com`, 2012.

[78] C. in Science. 2000 index, computing in science and engineering, volume 2. *Computing in Science and Engineering*, 2:94–97, 2000.

[79] F. P. Incropera. *Fundamentals of Heat and Mass Transfer*. John Wiley & Sons, 2006.

[80] Intel. Teraflop, research. `http://www.intel.com/content/www/us/en/research/intel-labs-teraflops-research-chip.html`.

[81] A. Ioffe. *Semiconductor thermoelements, and Thermoelectric cooling*. Infosearch, ltd., 1957.

[82] M. Ishikoa, M. Usuia, T. Ohuchib, and M. Shiraib. Design concept for wire-bonding reliability improvement by optimizing position in power devices. *7th International Seminar on Power Semiconductor*, 37:262–268, 2006.

[83] M. Jelodar, S. Fakhraie, F. Montazeri, S. Fakhraie, and M. Ahmadabadi. A representation for genetic-algorithm-based multiprocessor task scheduling. In *Evolutionary Computation, 2006. CEC 2006. IEEE Congress on*, pages 340 –347, 2006.

[84] S. K., S. R., H. W., V. S., S. K., and T. D. Temperature-aware microarchitecture. *SIGARCH Comput. Archit. News*, 31(2):2–13, May 2003.

[85] C. Kai-Yuan and W. D. F. Thermal placement for high-performance multichip modules. In *Proceedings of the 1995 International Conference on Computer Design: VLSI in Computers and Processors*, ICCD '95, pages 218–223, Washington, DC, USA, 1995. IEEE Computer Society.

[86] Karmarkar. A new polynomial-time algorithm for linear programming. *Combinatorica 4*, pages 373–395, 1984.

[87] D. H. Kim, K. Athikulwongse, and S. K. Lim. A study of through-silicon-via impact on the 3D stacked IC layout. In *Proceedings of the 2009 International Conference on Computer-Aided Design*, ICCAD '09, pages 674–680, New York, NY, USA, 2009. ACM.

[88] D. H. Kim, S. Mukhopadhyay, and S. K. Lim. Through-silicon-via aware interconnect prediction and optimization for 3D stacked ICs. In *Proceedings of the 11th international workshop on System level interconnect prediction*, SLIP '09, pages 85–92, New York, NY, USA, 2009. ACM.

[89] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by simulated annealing. *Science*, 220(4598):671–680, 1983.

[90] K. Kota, P. Hidalgo, Y. Joshi, and A. Glezer. Hybrid liquid immersion and synthetic jet heat sink for cooling 3-D stacked electronics. *Components, Packaging and Manufacturing Technology, IEEE Transactions on*, 2(5):817 –824, may 2012.

[91] E. A. G. Labs. Evans analytical group labs. `http://www.eaglabs.com/`.

[92] P. Lall, M. Pecht, and E. Hakim. *Influence of Temperature on Microelectronics and System Reliability*. The electronic packaging series. CRC PressINC, 1997.

[93] C. Lee, M. Liamini, and L. Frechette. A silicon microturbopump for a rankine-cycle power-generation microsystem. *Microelectromechanical Systems, Journal of*, 20(1):326 –338, feb. 2011.

[94] H. Lee and K. Chakrabarty. Test challenges for 3D integrated circuits. In *Design Test, IEEE*, pages 1–1, 2013.

[95] J. Lee, Y. I. Chang, and S. H. Chang. Cost benefit analysis of advanced nuclear fuel cycle using linear programming optimization. *Annals of Nuclear Energy*, 46(0):116 – 120, 2012.

[96] M. Li, X. Gong, W. Dong, and W. Li. The cooling tower fan system fault monitoring based on the trunk of RS485. In *Electronic and Mechanical Engineering and Information Technology (EMEIT), 2011 International Conference on*, volume 6, pages 3214 –3216, aug. 2011.

[97] X. Li, Y. Ma, and X. Hong. A novel thermal optimization flow using incremental floorplanning for 3D ICs. In *Design Automation Conference, 2009. ASP-DAC 2009. Asia and South Pacific*, pages 347 –352, jan. 2009.

[98] X. Li, Y. Ma, X. Hong, S. Dong, and J. Cong. LP based white space redistribution for thermal via planning and performance optimization in 3D ICs. In *Proceedings of the 2008 Asia and South Pacific Design Automation Conference*, ASP-DAC '08, pages 209–212, Los Alamitos, CA, USA, 2008. IEEE Computer Society Press.

[99] Y. Liang, K.-S. Leung, and K.-H. Lee. A novel binary variable representation for genetic and evolutionary algorithms. In *Evolutionary Computation, 2006. CEC 2006. IEEE Congress on*, pages 536 –543, 2006.

[100] J. H. Lienhard and J. H. Lienhard. *A Heat Transfer Textbook - 3rd ed.* Phlogiston Press: Cambridge, Massachusetts, 2008.

[101] S. K. Lim. 3D circuit design with through-silicon-via: Challenges and opportunities. EDPS 2010, 2010.

[102] J.-M. Lin, Y.-W. Chang, and S.-P. Lin. Corner sequence - a p-admissible floorplan representation with a worst case linear-time packing scheme. *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, 11(4):679–686, 2003.

[103] P. Lindner, V. Dragoi, T. Glinsner, C. Schaefer, and R. Islam. 3D interconnect through aligned wafer level bonding. In *Electronic Components and Technology Conference, 2002. Proceedings. 52nd*, pages 1439–1443, 2002.

[104] H. Ma, C. Liu, H. Su, and W. Ho. Study of a cooling system with a piezoelectric fan. In *Semiconductor Thermal Measurement and Management Symposium (SEMI-THERM), 2012 28th Annual IEEE*, pages 243 –248, march 2012.

[105] A. Materials. Applied materials. `http://www.appliedmaterials.com/tsv`.

[106] Matousek, Jiri, Gärtner, and Bernd. *Understanding and Using Linear Programming*. Springer, 8 edition, 2007.

[107] Mehrotra. On the implementation of a primal dual interior point method. *SIAM Journal on Optimization*, 2:575–601, 1992.

[108] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller. Equation of State Calculations by Fast Computing Machines. *The Journal of Chemical Physics*, 21(6):1087–1092, 1953.

[109] Q. Miao, M. Azarian, and M. Pecht. Cooling fan bearing fault identification using vibration measurement. In *Prognostics and Health Management (PHM), 2011 IEEE Conference on*, pages 1 –5, june 2011.

[110] A. Mitsos, I. N. Melas, P. Siminelakis, A. D. Chairakaki, J. Saez-Rodriguez, and L. G. Alexopoulos. Identifying drug effects via pathway alterations using an integer linear programming optimization formulation on phosphoproteomic data. *PLoS Comput Biol*, 5(12), 12 2009.

[111] G. E. Moore. Cramming more components onto integrated circuits. *Solid-State Circuits Society Newsletter, IEEE*, 11(5):33–35, 2006.

[112] F. Mulas, M. Pittau, M. Buttu, S. Carta, A. Acquaviva, L. Benini, and D. Atienza. Thermal balancing policy for streaming computing on multiprocessor architectures. In *Proceedings on DATE*, pages 734–739, 2008.

[113] S. Nakatake, K. Fujiyoshi, H. Murata, and Y. Kajitani. Module placement on bsg-structure and ic layout applications. In *Computer-Aided Design, 1996. ICCAD-96. Digest of Technical Papers., 1996 IEEE/ACM International Conference on*, pages 484–491, 1996.

[114] H. Nguyen, I. Yoshihara, and M. Yasunaga. Modified edge recombination operators of genetic algorithms for the traveling salesman problem. In *Industrial Electronics Society, 2000. IECON 2000. 26th Annual Confjerence of the IEEE*, volume 4, pages 2815 –2820 vol.4, 2000.

[115] B. Noia and K. Chakrabarty. Testing and design-for-testability techniques for 3D integrated circuits. In *Test Symposium (ATS), 2011 20th Asian*, pages 474–479, 2011.

[116] V. Nollet, P. Avasare, J.-Y. Mignolet, and D. Verkest. Low cost task migration initiation in a heterogeneous MP-SoC. In *Proceedings of the Conference on DATE*, pages 252–253, 2005.

[117] U. Ogras, R. Marculescu, P. Choudhary, and D. Marculescu. Voltage-frequency island partitioning for GALS-based networks-on-chip. In *Design Automation Conference, 2007. DAC '07. 44th ACM/IEEE*, pages 110 –115, june 2007.

[118] N. Okada, C. Kodama, T. Sato, and K. Fujiyoshi. Thermal driven module placement using sequence-pair. In *Circuits and Systems, 2006. APCCAS 2006. IEEE Asia Pacific Conference on*, pages 1871 –1874, dec. 2006.

[119] I. Ono, M. Yamamura, and S. Kobayashi. A genetic algorithm for job-shop scheduling problems using job-based order crossover. In *Evolutionary Computation, 1996., Proceedings of IEEE International Conference on*, pages 547 –552, may 1996.

[120] G. Paci, P. Marchal, F. Poletti, and L. Benini. Exploring temperature-aware design in low-power MPSoCs. In *Proceedings of the DATE*, volume 1, pages 1–6, March 2006.

[121] R. Patel and M. Raghuwanshi. Review on real coded genetic algorithms used in multiobjective optimization. In *Emerging Trends in Engineering and Technology (ICETET), 2010 3rd International Conference on*, pages 610 –613, nov. 2010.

[122] H. N. Phan and D. Agonafer. Experimental analysis model of an active cooling method for 3D-ICs utilizing multidimensional configured thermoelectric coolers. *Journal of Electronic Packaging*, 132(2), 2010.

[123] D. Puschini, F. Clermidy, P. Benoit, G. Sassatelli, and L. Torres. Temperature-aware distributed run-time optimization on MP-SoC using game theory. In *IEEE Computer Society Annual Symposium on VLSI*, 2008.

[124] K. Puttaswamy and G. H. Loh. 3D-Integrated SRAM components for high-performance microprocessors. *IEEE Transactions on Computers*, 58(10):1369–1381, 2009.

[125] I. Radu, D. Landru, G. Gaudin, G. Riou, C. Tempesta, F. Letertre, L. Di Cioccio, P. Gueguen, T. Signamarcheix, C. Euvrard, J. Dechamp, L. Clavelier, and M. Sadaka. Recent developments of Cu-Cu non-thermo compression bonding for wafer-to-wafer 3D stacking. In *3D Systems Integration Conference (3DIC), 2010 IEEE International*, pages 1–6, 2010.

[126] J. Rawlings and R. Amrit. Optimizing process economic performance using model predictive control. In L. Magni, D. Raimondo, and F. Allgawer, editors, *Nonlinear Model Predictive Control*, volume 384 of *Lecture Notes in Control and Information Sciences*, pages 119–138. Springer Berlin / Heidelberg, 2009.

[127] J. L. Risco. Java evolutionary computation. `http://sourceforge.net/projects/jeco/`, 2012.

[128] R. Rutenbar. Simulated annealing algorithms: an overview. *Circuits and Devices Magazine, IEEE*, 5(1):19 –26, jan. 1989.

[129] M. Sabry, D. Atienza, and A. Coskun. Thermal analysis and active cooling management for 3D MPSoCs. *Circuits and Systems (ISCAS)*, pages 2237–2240, 2011.

[130] M. Salinas. 3-D thermoelectric cooler analysis. In *Semiconductor Thermal Measurement and Management Symposium, 2000. Sixteenth Annual IEEE*, pages 10 –18, 2000.

[131] K. Sankaranarayanan, S. Velusamy, M. R. Stan, and K. Skadron. A case for thermal-aware floorplanning at the microarchitectural level. *Journal of Instruction Level Parallelism*, 7:8–16, 2005.

[132] S. Saponara and L. Fanucci. Homogeneous and heterogeneous MPSoC architectures with network-on-chip connectivity for low-power and real-time multimedia signal processing. *VLSI Design*, 2012(450302):1–17, 2012.

[133] R. Shah and A. London. *Laminar flow forced convection in ducts: a source book for compact heat exchanger analytical data*. Advances in heat transfer : Supplement. Academic Press, 1978.

[134] L. Shang, L.-S. Peh, A. Kumar, and N. K. Jha. Thermal modeling, characterization and management of on-chip networks. In *Proceedings of the 37th annual IEEE/ACM International Symposium on Microarchitecture*, MICRO 37, pages 67–78, Washington, DC, USA, 2004. IEEE Computer Society.

[135] B. Shi and A. Srivastava. TSV-constrained micro-channel infrastructure design for cooling stacked 3D-ICs. In *Proceedings of the 2012 ACM international symposium on International Symposium on Physical Design*, ISPD '12, pages 113–118. ACM, 2012.

[136] V. Singh and S. Choudhary. Genetic algorithm for traveling salesman problem: Using modified partially-mapped crossover operator. In *Multimedia, Signal Processing and Communication Technologies, 2009. IMPACT '09. International*, pages 20 –23, march 2009.

[137] V. Singhal, S. V. Garimella, and A. Raman. Microscale pumping technologies for microchannel cooling systems. *Applied Mechanics Reviews*, 57(3):191–221, 2004.

[138] S. N. Sivanandam and S. N. Deepa. *Introduction to Genetic Algorithms*. Springer, 2007.

[139] K. Skadron, M. R. Stan, K. Sankaranarayanan, W. Huang, S. Velusamy, and D. Tarjan. Temperature-aware microarchitecture: modeling and implementation. *Trans. Architecture Code Optimizations 1*, pages 94–125, 2004.

[140] A. Sridhar, A. Vincenzi, M. Ruggiero, T. Brunschwiler, and D. Atienza. 3D-ICE: fast compact transient thermal modeling for 3D ICs with inter-tier liquid cooling. In *Proceedings of the International Conference on Computer-Aided Design*, ICCAD '10, pages 463–470, 2010.

[141] P. Stravers. Homogeneous multiprocessing for the masses. In *Embedded Systems for Real-Time Multimedia, 2004. ESTImedia 2004. 2nd Workshop on*, page 3, sept. 2004.

[142] H. Su, F. Liu, A. Devga, E. Acar, and S. Nassif. Full chip leakage estimation considering power supply and temperature variations. *Proceeding of the ISPD*, pages 78–83, 2003.

[143] T. T. Y. Suen and J. S. K. Wong. Efficient task migration algorithm for distributed systems. *IEEE Transactions on Parallel and Distributed Systems*, 3(4):488–499, July 1992.

[144] G. SV and S. CB. Transport in microchannels: A critical review. *Ann. Rev. Heat Transfer, 2003*, 2003.

[145] D. Sylvester and C. Wu. Analytical modeling and characterization of deep-submicrometer interconnect. *Proceedings of the IEEE*, 89(5):634–664, 2001.

[146] M. Tang and X. Yao. A memetic algorithm for VLSI floorplanning. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 37(1):62 –69, feb. 2007.

[147] C.-H. Tsai. Cell-level placement for improving substrate thermal distribution. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 19(2):253–266, Feb. 2000.

[148] Tsunekane and Taira. Design and performance of compact heatsink for high-power diode edge-pumped, microchip lasers. *Selected Topics in Quantum Electronics, IEEE Journal of*, 13(3):619 –625, may-june 2007.

[149] P. G. D. Valle and D. Atienza. Emulation-based transient thermal modeling of 2D/3D systems-on-chip with active cooling. *Microelectronics Journal*, 1:564–571, 2010.

[150] A. Viswanath, W. Fang, X. Zhang, V. Ganesh, and L. Lim. Numerical analysis by 3D finite element wire bond simulation on Cu/low-k structures. In *Electronic Packaging Technology Conference, 2005. EPTC 2005.*, 2005.

[151] J. Vlach and K. Singhal. *Computer methods for circuit analysis and design*. Springer, 1983.

[152] C. Wan, J. Wang, G. Yang, X. Li, and X. Zhang. Optimal micro-siting of wind turbines by genetic algorithms based on improved wind and turbine models. In *Decision and Control, 2009 held jointly with the 2009 28th Chinese Control Conference. CDC/CCC 2009. Proceedings of the 48th IEEE Conference on*, pages 5092 –5096, dec. 2009.

[153] D. Wang, J. Knighten, and P. Muller. An integrated vent, heatsink and EMI shield. In *Semiconductor Thermal Measurement and Management, 2002. Eighteenth Annual IEEE Symposium*, pages 125 –131, march 2002.

[154] D. Weile and E. Michielssen. Integer coded pareto genetic algorithm design of constrained antenna arrays. *Electronics Letters*, 32(19):1744 –1745, sep 1996.

[155] E. Wong and S. K. Lim. 3D floorplanning with thermal vias. In *Proceedings of the conference on Design, automation and test in Europe: Proceedings*, DATE '06, pages 878–883, 3001 Leuven, Belgium, Belgium, 2006. European Design and Automation Association.

[156] I. K. Y. Han and C. Moritz. Temperature aware floorplanning. In *Workshop on Temperature-Aware Computer Systems*, 2005.

[157] J. Yang, X. Zhou, M. Chrobak, Y. Zhang, and L. Jin. Dynamic thermal management through task scheduling. In *Proceedings of the IEEE International Symposium on Performance Analysis of Systems and software*, pages 191–201, 2008.

[158] I. Yeo and E. J. Kim. Temperature-aware scheduler based on thermal behavior grouping in multicore systems. In *Proceedings of the Conference on DATE*, 2009.

[159] M. Yoshikawa and H. Terai. Hybrid genetic algorithm engine for high-speed floorplanning. In *Circuit Theory and Design, 2005. Proceedings of the 2005 European Conference on*, pages I/189 – I/192 vol. 1, aug.-2 sept. 2005.

[160] A. M. Young and S. J. Koester. *Three Dimensional Integrated Circuit Design (chapter 1: 3D Process Technology Considerations)*. Integrated Circuits and Systems, 2010.

[161] E. Young, C. Chu, and Z. Shen. Twin binary sequences: a nonredundant representation for general nonslicing floorplan. *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, 22(4):457–469, 2003.

[162] D. Zhang and J.-Q. Lu. Evaluation of fabrication process for a novel chip-to-wafer (C2W) 3D integration approach using an alignment template. In *Advanced Semiconductor Manufacturing Conference (ASMC), 2012 23rd Annual SEMI*, pages 398–403, 2012.

[163] H. Zhou and J. Wang. Acg-adjacent constraint graph for general floorplans. In *Computer Design: VLSI in Computers and Processors, 2004. ICCD 2004. Proceedings. IEEE International Conference on*, pages 572–575, 2004.